FORMAL COMPUTER ORIENTED STIFFNESS DESIGN METHOD FOR LARGE AND COMPLEX BUILDING FRAMES

A THESIS

Presented to

The Faculty of the Division of Graduate

Studies and Research

By

Barry Louis Adelson

In Partial Fulfillment

of the Requirements for the Degree

Master of Science in Civil Engineering

Georgia Institute of Technology

January 1977

FORMAL COMPUTER ORIENTED STIFFNESS DESIGN METHOD

FOR LARGE AND COMPLEX BUILDING FRAMES

Approved:

Leroy Z. Emkin, Chairman

Barry J. Goodno

Kenneth M. Will

Date approved by Chairman: 1/18/77

ACKNOWLEDGEMENTS

The author wishes to thank Dr. Leroy Emkin, Thesis advisor, for his guidance and assistance in completing this project, as well as Dr. Barry J. Goodno and Dr. Kenneth M. Will for serving on the reading committee.

The work reported herein was conducted under National Science Foundation Research Initiation Grant No. ENG74-09797 A01 (formerly GK-42131) and this support is greatly appreciated.

Computer related work was performed on a CDC CYBER 74/28 located at the Georgia Tech Computer Center, and an IBM 370/158 located at the University of Georgia Computer Center in Athens, Georgia. Access to both computers was through a COPE 1620 Remote Job Entry Terminal located in the Civil Engineering Systems Laboratory, Georgia Institute of Technology.

Expert advice and assistance given by Mr. Mark A. Prichard, Research Engineer, and Mr. David B. Green, Assistant Research Engineer, are gratefully acknowledged.

The architectural/engineering firm of John Portman & Associates, Atlanta, Georgia, is acknowledged for their high quality engineering environment which provided many insights into the performance of high rise buildings.

Special thanks are due Ms. Margaret Vogt and Ms. Elizabeth Gray for their expert and professional typing of this thesis.

Appreciation is expressed to the Division of Graduate Studies for waiving certain format requirements, so that this thesis might also serve as the final report for National Science Foundation Research Grant No. ENG74-09797 A01.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

CHAPTER 1

INTRODUCTION

The overall process of structural design includes every step from the original determination of the need for a facility through the final drawings, specifications, and supervision of construction. The creative part of the design, however, occurs early in the decision-making process and consists of selection of the overall system configuration, materials, and the location, arrangements, and sizing of components. This segment of the decision-making process obviously has a dominating influence on the resulting aesthetics, utility, and economy of the project. In the past, the creative part of the design has been primarily based on the designer's intuition and past experience. Although there certainly cannot be a substitute for experience, the engineering use of design-oriented computer programs on the digital computer has made possible improvements in both the creative part of the design process and also the remaining design sequence.

In the structural design of modern multistory buildings, computers are taking on ever increasing importance. It is a curious development, however, that their use in the structural engineering design office environment has primarily been to analyze rather than design. The concept of structural design as used herein refers to that process wherein the sizing and location of structural components is based not only on an individual component's behavior, but is also based on the effect the component has on the overall structure's performance characteristics. Where computers are used for design purposes, they are almost exclusively used to design on the basis of element strength, stiffness, and stability requirements, and do not specifically consider the effect on overall frame stiffness. However, with the increasing height to width ratio of multistory structures, the increasing use of higher strength steels, and the decreasing use of heavy cladding, partitions, walls, and other similar components, the satisfaction of strength constraints alone is not sufficient for a well designed structure. The reason is that strength design under these conditions result in less stiff torsional building displacements. In fact, in these cases, design for stiffness generally becomes more critical than design for strength.

It is recognized that design based on limiting displacements to be within acceptable levels is not necessarily sufficient. In particular, as is reported in several papers (4,10,12)[1], the more important effect with regard to human comfort in buildings is acceleration level, whereas large displacements have more impact on performance of windows, partitions, walls, cladding, and the like. In order to limit building accelerations, however, design procedures must specifically consider mass distribution and damping in addition to stiffness.

---

[1]Numbers in parentheses refer to references given in the REFERENCES Section.

Although such procedures are desirable and need to be researched, their application would be helpful only in those circumstances where acceleration computations (i.e., dynamic analysis) were performed. However, the great majority of tall buildings in the U.S. do not receive such dynamic analysis. Acceleration design, if performed, would also include a form of stiffness design, and no formal stiffness design procedures for large multistory building frames are reported in the literature.

Therefore, the purpose of this project was to develop a formal computer-oriented stiffness design procedure for large and complex plane and space frame building structures in which cladding, partition walls, and other 'nonstructural' components are neglected with respect to their stiffness contributions, and in which the use of shear walls to provide substantial lateral stiffness is nonexistent. In addition, although substantial benefit to stiffness may be obtained by manipulating a frame's geometrical configuration (5,14) the method developed is intended to be limited to the majority of cases where architectural, aesthetic, and other requirements make such geometrical changes infeasible. Finally, the project develops a stiffness design procedure based on minimizing material weight. It might be argued that material weight interpreted in terms of cost represents such a small percent of the total building cost, that weight minimization alone is inappropriate. While this may have been a valid argument in the past, current world problems with respect to available natural resources dictate that material conservation in all aspects of engineering design must take on new dimensions of importance and, in view of this, material weight minimization is justified.

Furthermore, in present office practice, considerable effort is devoted to generating fairly sophisticated strength designs, while design for frame stiffness is performed on a rule-of-thumb, past experience basis. Recognizing that experience is certainly the mainstay of the practicing structural engineer, it seems somewhat inconsistent to use formal design procedures to satisfy strength requirements which do not control design, while at the same time using highly informal procedures to satisfy stiffness requirements which are the controlling design considerations for most tall buildings. The application of such informal procedures, although capable of satisfying deflection limitations, are much too unreliable with respect to efficient material utilization in tall buildings.

The literature is strikingly void of descriptions of design procedures for complex frames where the controlling design criteria are deflection limitations. Those procedures that are mentioned are so limited in scope that their application in a design office environment is not practical.

Some reported studies are as follows. Moses (11)investigated the least-weight design problem of a three-bar truss and a one-story one-bay rigid frame using the cutting-plane method of nonlinear programming. Brown and Ang (2) utilized a gradient projection method of nonlinear programming in the minimum weight design of one and two-story one-bay rigid frames subject to both stress and deflection constraints. Romstad and Wang (13) formulated the minimum weight problem of elastic framed structures as a linear programming problem. Kavlie and Moe (9) utilized suboptimization schemes in conjunction with a sequential unconstrained minimization technique in the solution to a nonlinear programming formulation of the optimization of statically indeterminate frames. Cella and Logcher (3) apply a branch-and-bound algorithm in the discrete variable nonlinear optimization of linear elastic framed structures.

All of the thse studies, as well as several other reported studies, represent significant and important contributions to the area of mathematical programming techniques as applied to the structural design process. Although these studies result in new techniques for optimum design of framed structures, none of them were shown to be effective, practical, efficient, economical, or even applicable when applied to large scale, complex multistory plane and space structures, especially when deflection limitation was a controlling factor.

The principle objective of this project was to develop a computer-oriented, rational, efficient, and immediately applicable method of elastic stiffness design of general building frames for lateral loads. Space framed structures of arbitrary geometric configuration were considered. The method makes no attempt to alter specified frame geometry. The design method satisfies up to three arbitrarily specified building deflection constraints specified in terms of maximum absolute displacements. The material type of the member elements are limited to steel. Member selection is based upon a discrete set of available section sizes, input to the design system, such as is contained in the 1970 American Institute of Steel Construction's Manual of Steel Construction (1) section tables.

The basis of the stiffness design method is a heuristic gradient search optimization algorithm intended to minimize material weight (cost) increases in order to satisfy up to three imposed building deflection constraints. The method involves three stages which are an exact stiffness analysis of the frame utilizing any user provided analysis program appropriately modified (Chapter 4), a kinematic condensation analysis of the frame (Chapter 3), and the heuristic gradient search optimization procedure which uses a virtual work approximation for displacement computation (Chapter 2). As will be discussed in Chapter 3, the kinematic condensation procedure is not recommended for use due to its resulting inefficiency relative to standard stiffness analysis in the context of this design system. Consequently, only the modified user supplied program is recommended for exact analysis of the frame in this design system.

Chapters 2 and 3 describe the theoretical basis of the heuristic gradient search design optimization and kinematic condensation procedures, respectively. Chapter 4 discusses the required modifications to a user supplied stiffness analysis program. Chapter 5 discusses several example design problems. Chapter 6 discusses important conclusions and recommendations resulting from this study. And finally, the Appendices contain computer program documentation, listings, and other important details.

CHAPTER 2

DISPLACEMENT CONSTRAINT DESIGN

2.1 Introduction

The proposed stiffness design method is based on assumptions of linear elastic behavior, and is specifically intended to satisfy imposed lateral deflection constraints at selected joints in the structure. Maximum joint deflections are specified for certain applied lateral loadings. Actual joint displacements are calculated either by a rigorous exact analysis or by the approximate method described in Chapter 3. Should these computed displacements be greater than the specified maximum, then various members in the frame are increased in size using a heuristic gradient search optimization technique intended to minimize frame cost increases while reducing displacements in order to satisfy the displacement constraints.

Strength and stability design is not performed in this method. It is recommended that a design satisfying strength and stability conditions are satisfied prior to executing stiffness design procedures described herein. It should also be noted that all members are assumed to be prismatic.

2.2 Gradient Search Procedure

A maximum of three displacement constraints may be specified for a frame. If one or more of these constraints are exceeded, the heuristic gradient search optimization procedure is executed to reduce the computed displacements below the specified maximum.

The procedure is based upon the calculation of displacement sensitivity coefficients, Q_i, for each member in the frame and corresponding to each displacement constraint which is exceeded. Q_i is defined as the decrease in some displacement component (i.e. the one which is being decreased to satisfy a displacement constraint) due to an increase in cost of a member. After all Q_i's are computed for each member in the frame (i.e. one value for each member, for each displacement constraint and for each applied lateral load condition), the procedure simply involves selecting the member with the most negative value of Q_i as the next candidate member to increase in size. This member represents the least increase in cost of the frame required to decrease the displacement in question by a unit amount.

After a particular member is increased in size (by one section size in a user provided section table), its Q_i's must be recomputed since Q_i is a non-linear function of the member's area (see Section 2.3).

Now, as will be shown in Section 2.3, Q_i is also a function of the forces in a member. So, if one frame member increases in size, the result would be force changes throughout the entire structure, thereby requiring a new analysis to determine these force changes, followed by a recomputation of all other Q_i's. However, such a procedure would make the stiffness design prohibitively costly. Consequently, in order to allow a cost-effective stiffness design procedure, it is assumed that force changes associated with each member size change are small and may be assumed not to occur. Only the effect of a member size change on displacement is recomputed in a very efficient, although approximate, way using the Unit Load Method of Virtual Work where only the term associated with a single member size change need be recomputed since its properties have changed, but where its internal forces are assumed not to have changed.

After the Q_i's for the changed member are recomputed, and the effect on the displacements are determined, the next most negative Q_i is selected for the next member size increase. This process is repeated until all displacement constraints are satisfied under all specified loading conditions. At this point, a more exact computation of the frame displacements is performed, also resulting in an update of all member forces. If the "exact" displacements still exceed the displacement constraints, then the entire process is repeated. A final design for stiffness is found when all displacement constraints are satisfied after an "exact" displacement analysis. The details of this process are described in Section 2.5.

2.3 Formulation of the Displacement Sensitivity Coefficient

Consider the cost, f, of all members in a structure which affect displacements,

$$f = \sum_{i=1}^{M} (u_i \rho_i L_i A_i) \tag{2.1}$$

where,

u_i = unit material cost of member i

ρ_i = the mass density of member i

L_i = length of member i

A_i = area of member i

M = number of members in the structure

Because it is desired to determine the increase in material cost, f, due to a unit decrease in joint deflection, Δ , Eq. 2.1 is differentiated with respect to Δ as,

$$\frac{\partial f}{\partial \Delta} = \sum_{i=1}^{M} u_i \rho_i L_i \frac{\partial A_i}{\partial \Delta} \qquad (2.2)$$

Since a member is increased in size, cost will increase, but the larger member size decreases deflection. Therefore, ∂f is positive while $\partial \Delta$ is negative and $\frac{\partial f}{\partial \Delta}$ is negative. It is desired to have a small increase in cost for a large decrease in deflection, so $\frac{\partial f}{\partial \Delta}$ must be maximized (i.e. made least negative or minimize its magnitude). It now becomes convenient to define a deflection sensitivity coefficient, Q_i, as follows ($\frac{\partial \Delta}{\partial A_i}$ is derived in Section 2.4),

$$Q_i = \frac{1}{u_i \rho_i L_i} \frac{\partial \Delta}{\partial A_i} \qquad (2.3)$$

Substituting Q_i into Eq. 2.2, results in,

$$\frac{\partial f}{\partial \Delta} = \sum_{i=1}^{M} \left(\frac{1}{Q_i} \right) \qquad (2.4)$$

The procedures used to maximize (i.e. make least negative) $\frac{\partial f}{\partial \Delta}$ operate on a member by member basis. The member with the least negative value of $\frac{1}{Q_i}$, or most negative value of Q_i, is selected to increase in size next, followed by a recomputation of the member's Q_i and the new displacements. This process is repeated until displacement constraints are satisfied.

The procedure is analogous to moving along the gradient $\frac{\partial f}{\partial \Delta}$, of the cost function, f, in an incremental fashion, member by member, until a final design is found which represents the least increase in cost of a frame required to satisfy the imposed displacement constraints.

Now, to calculate Q_i for a member by Eq. 2.3, the parameters u_i, ρ_i, and L_i are known at the start and remain constant. However, the factor $\frac{\partial \Delta}{\partial A_i}$ must be computed at the start and also each time the member changes size. This factor is developed next.

2.4 Computation of $\partial \Delta / \partial A_i$ and Final Form of Q_i

Since the factor $\frac{\partial \Delta}{\partial A_i}$ is computed often, it is necessary to have an extremely fast method for its computation. The method selected here is based on the Unit Load Method of Virtual Work.

At the beginning of the stiffness design method, an exact frame analysis is performed for the various applied lateral joint load conditions corresponding to the displacement constraints. Included in the lateral load conditions are unit load conditions, one corresponding to each displacement constraint (i.e. a unit load placed at a joint and in a direction for which a displacement constraint is specified). It is important to note that lateral deflections associated with gravity loads applied to members are specifically neglected in order to make the evaluation of the virtual work equation as efficient as possible. This is not considered to have a significant effect on the final results. However, a less restricting procedure could be used for gravity loads to approximately account for sway displacements by placing all gravity loads on the frame as vertical joint loads. Only member loads will not be considered. In addition, the effects of shear and torsion deformations of members are also neglected, so that only biaxial bending and axial member deformations are accounted for in the virtual work equation.

The resulting virtual work equation to compute the displacement corresponding to a displacement constraint is therefore,

$$\Delta_{VW} = \sum_{i=1}^{M} \frac{F_{Qi} F_{Pi} L_i}{A_i E_i} + \sum_{i=1}^{M} \int_0^{L_i} \frac{M^y_{Qi}(x) M^y_{Pi}(x)}{E_i I_{yi}} dx_i$$

$$+ \sum_{i=1}^{M} \int_0^{L_i} \frac{M^z_{Qi}(x) M^z_{Pi}(x)}{E_i I_{zi}} \qquad (2.5)$$

where,

F_{Pi} = axial force in member i due to applied lateral load

F_{Qi} = axial force in member i due to unit virtual load

A_i = cross sectional area of member i

E_i = modulus of elasticity of member i

L_i = length of member i

$M^y_{Pi}(x)$ = local y-axis bending moment as it varies along the length (x) of member i, resulting from applied lateral load

$M^y_{Qi}(x)$ = local y-axis bending moment as it varies along the length (x) of member i, resulting from unit virtual load

I_{yi} = moment of inertia about the local y-axis of member i

$M^z_{Pi}(x)$ = local z-axis bending moment as it varies along the length (x) of member i resulting from applied lateral load

$M^z_{Qi}(x)$ = local z-axis bending moment as it varies along the length (x) of member i, resulting from unit virtual load

I_{zi} = moment of inertia about local z-axis of member i

Although the exact displacement corresponding to each displacement constraint is known at the beginning of the stiffness design process from the first exact analysis, the virtual work equation, Eq. 2.5, will be used to compute the new displacement resulting from a change in member size. This will be done by taking the difference between a selected member's contribution to Δ_{VW}, say $\Delta_{VW,i}$, at its current size and its contribution to Δ_{VW} at its new size and subtracting the result, which represents the change in displacement due to a member size increase, from the current value of displacement, which at the start of the process would be the exact displacement value computed from the exact analysis.

Now, the contribution of member i to Δ_{VW} is given by,

$$\Delta_{VW,i} = \frac{F_{Qi}F_{Pi}L_i}{A_iE_i} + \int_0^{L_i} \frac{M^y_{Qi}(x)M^y_{Pi}(x)}{E_iI_{yi}}\,dx_i + \int_0^{L_i} \frac{M^z_{Qi}(x)M^z_{Pi}(x)}{E_iI_{zi}}\,dx_i \qquad (2.6)$$

For the loading conditions considered in this method, i.e. lateral loads applied at joints only, and gravity loads applied along the length of members neglected (gravity loads may be applied as vertical joint loads), the moment diagrams along the lengths of members are all linear functions of the centroidal distance x_i along the member. Consequently, the integrals in Eq. 2.6 can be evaluated directly in terms of the proper member end moments.

So, consider member i going from joint j to joint k, and bending moment diagrams for applied joint loads P and applied unit virtual loads Q for both local y and z principal bending axes of the member as shown in Fig. 2.1. The form of each moment diagram is mx + b, where m is the slope, b is the moment at the start of the member (at joint j), and x is the distance along the centroidal axis of the member measured from the start of the member.

The notation used in Fig. 2.1 is defined as follows,

MEA2 = moment about local y-axis at j end of member i resulting from applied lateral load

MEA3 = moment about local z-axis at j end of member i resulting from applied lateral load

MEA5 = moment about local y-axis at k end of member i resulting from applied lateral load

MEA6 = moment about local z-axis at k end of member i resulting from applied lateral load

MEAU2 = moment about local y-axis at j end of member i resulting from unit virtual load

MEAU3 = moment about local z-axis at j end of member i resulting from unit virtual load

MEAU5 = moment about local y-axis at k end of member i resulting from unit virtual load

MEAU6 = moment about local z-axis at k end of member i resulting from unit virtual load

$YA = \dfrac{MEA5 - MEA2}{L_i}$ = Slope of y-axis moment diagram for member i resulting from applied lateral load

$YC = \dfrac{MEAU5 - MEAU2}{L_i}$ = Slope of y-axis moment diagram for member i resulting from unit virtual load

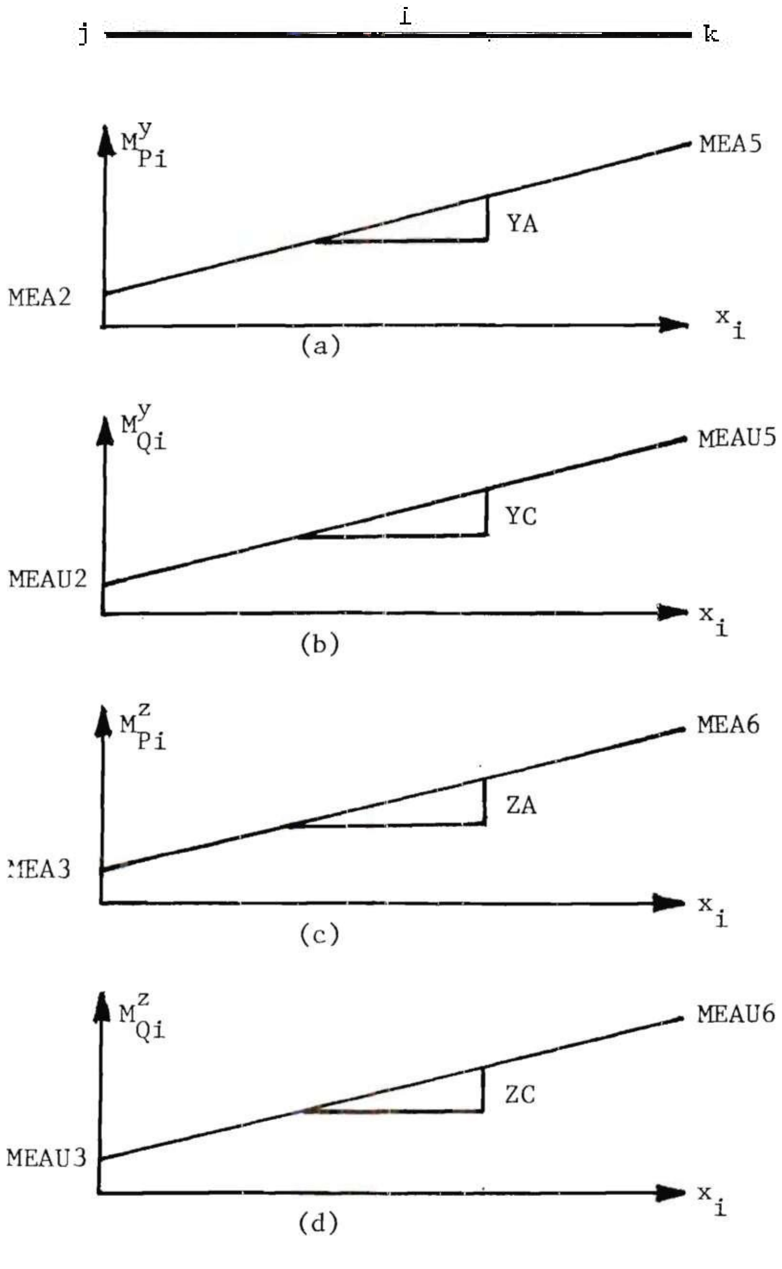


Figure 2.1 Moment Diagrams for Member i

$ZA = \dfrac{MEA6 - MEA3}{L_i}$ = Slope of z-axis moment diagram for member i resulting from applied lateral load

$ZC = \dfrac{MEAU6 - MEAU3}{L_i}$ = Slope of z-axis moment diagram for member i resulting from unit virtual load

The y-axis bending contribution to deflection of member i may now be assessed. From Eq. 2.6 this is represented as,

$$\Delta^y_{VW,i} = \int_0^{L_i} \frac{M^y_{Pi}(x) M^y_{Qi}(x)}{E_i I_{yi}} dx_i \tag{2.7}$$

The expression $M^y_{Pi}(x)$ is the equation of the straight line in Fig. 2.1(a) and is,

$$M^y_{Pi}(x) = MEA2 + (YA*x_i) \tag{2.8}$$

Figure 2.1(b) shows the straight line which represents $M^y_{Qi}(x)$ and is,

$$M^y_{Qi}(x) = MEAU2 + (YC*x_i) \tag{2.9}$$

Substituting Eqs. 2.8 and 2.9 into Eq. 2.7 results in,

$$\Delta^y_{VW,i} = \int_0^{L_i} \frac{YA*YC*x_i^2 + (MEA2*YC+YA*MEAU2)x_i + MEA2*MEAU2}{E_i I_{yi}} dx_i \tag{2.10}$$

Performing the required integration leads to,

$$\Delta^y_{VW,i} = \left. \frac{\dfrac{YA*YC*x_i^3}{3} + \dfrac{(MEA2*YC+YA*MEAU2)x_i^2}{2} + (MEA2*MEAU2)x_i}{E_i I_{yi}} \right]_0^{L_i} \tag{2.11}$$

and substituting in the limits results in,

$$\Delta^{y}_{VW,i} = \frac{\frac{YA*YC*L_i^3}{3} + \frac{(MEA2*YC+YA*MEAU2)L_i^2}{2} + (MEA2*MEAU2)L_i}{E_i I_{yi}} \tag{2.12}$$

It is now convenient to redefine the numerator of the Eq. 2.12 as,

$$B_{yi} = \frac{YA*YC*L_i^3}{3} + \frac{(MEA2*YC+YA*MEAU2)L_i^2}{2} + (MEA2*MEAU2)L_i \tag{2.13}$$

Eq. 2.12 now becomes,

$$\Delta^{y}_{VW,i} = \frac{B_{yi}}{E_i I_{yi}} \tag{2.14}$$

The moments shown in Figure 2.1 are all shown as positive for illustration purposes only. The derivation is valid for negative moments as well as positive ones.

Similarly, the relationships for $M^{z}_{Pi}(x)$ and $M^{z}_{Qi}(x)$ may be obtained from Figs. 2.1(c) and 2.1(d). When the necessary operations are performed, B_{zi} becomes,

$$B_{zi} = \frac{ZA*ZC*L_i^3}{3} + \frac{(MEA3*ZC+YA*MEAU3)L_i^2}{2} + (MEA3*MEAU3)L_i \tag{2.15}$$

and the contribution of z-axis bending of member i to deflection Δ_{VW} is,

$$\Delta^{z}_{VW,i} = \frac{B_{zi}}{E_i I_{zi}} \tag{2.16}$$

The total contribution to deflection of member i is therefore (from

Eqs. 2.6, 2.14, and 2.16),

$$\Delta_{VW,i} = \frac{F_{Pi}F_{Qi}L_i}{A_iE_i} + \frac{B_{yi}}{E_iI_{yi}} + \frac{B_{zi}}{E_iI_{zi}} \tag{2.17}$$

It is required to find the incremental change in deflection for an incremental change in area of member i. Consequently, differentiating Eq. 2.5 with respect to A_i leads to,

$$\frac{d\Delta_{VW}}{dA_i} = \frac{d\Delta_{VW,i}}{dA_i} \tag{2.18}$$

where from Eq. 2.17,

$$\frac{d\Delta_{VW,i}}{dA_i} = \frac{d}{dA_i}\left(\frac{F_{Pi}F_{Qi}L_i}{A_iE_i}\right) + \frac{d}{dA_i}\left(\frac{B_{yi}}{E_iI_{yi}}\right) + \frac{d}{dA_i}\left(\frac{B_{zi}}{E_iI_{zi}}\right) \tag{2.19}$$

As was previously stated, it is assumed that member forces and end moments (and therefore B_{yi} and B_{zi}) are constant during member size changes. Eq. 2.19 is then easily evaluated since only the moments of inertia and area change as the member area is increased. So, rearranging Eq. 2.19 results in,

$$\frac{d\Delta_{VW}}{dA_i} = \frac{d\Delta_{VW,i}}{dA_i} = \frac{F_{Pi}F_{Qi}L_i}{E_i}\frac{d}{dA_i}\left(\frac{1}{A_i}\right) + \frac{B_{yi}}{E_i}\frac{d}{dA_i}\left(\frac{1}{I_{yi}}\right) + \frac{B_{zi}}{E_i}\frac{d}{dA_i}\left(\frac{1}{I_{zi}}\right) \tag{2.20}$$

The derivative $\frac{d}{dA_i}(\frac{1}{A_i})$ is computed easily as,

$$\frac{d}{dA_i}\left(\frac{1}{A_i}\right) = -\frac{1}{A_{i,old}^2} \tag{2.21}$$

where,

$A_{i,old}$ = current cross-sectional area of member i

However, the derivatives of $(\frac{1}{I_i})$ are not so easily obtained since a general expression for I_i as a function of A_i is not available. Although an approximate expression could be developed, a more direct approach was used as follows,

$$\frac{d}{dA_i}\left(\frac{1}{I_{yi}}\right) = \frac{d(I_{yi})}{dA_i} \cdot \frac{d}{dI_{yi}}\left(\frac{1}{I_{yi}}\right)$$

$$= -\frac{1}{I_{yi,old}^2} \cdot \frac{I_{yi,new} - I_{yi,old}}{A_{i,new} - A_{i,old}} \tag{2.22}$$

and similarly,

$$\frac{d}{dA_i}\left(\frac{1}{I_{zi}}\right) = -\frac{1}{I_{zi,old}^2} \cdot \frac{I_{zi,new} - I_{zi,old}}{A_{i,new} - A_{i,old}} \tag{2.23}$$

where,

$A_{i,old}$ = current cross-sectional area of member i

$I_{yi,old}$ = current local y-axis moment of inertia of member i

$I_{zi,old}$ = current local z-axis moment of inertia of member i

$A_{i,new}$ = cross-sectional area of a section which is one larger than that of member i, in the appropriate section table

$I_{yi,new}$ = moment of inertia about local y-axis of a section which is one larger than that of member i, in the appropriate section table

$I_{zi,new}$ = moment of inertia about local z-axis of a section which is one larger than that of member i, in the appropriate section table

Now substitute Eqs. 2.21, 2.22 and 2.23 into Eq. 2.20,

$$\frac{d\Delta_{VW}}{dA_i} = \frac{-F_{Pi}F_{Qi}L_i}{A_i{}^2{}_{,old}E_i} - \frac{B_{yi}}{E_i I_{yi}{}^2{}_{,old}}\left(\frac{I_{yi,new} - I_{yi,old}}{A_{i,new} - A_{i,old}}\right)$$

$$- \frac{B_{zi}}{E_i I_{zi}{}^2{}_{,old}}\left(\frac{I_{zi,new} - I_{zi,old}}{A_{i,new} - A_{i,old}}\right) \qquad (2.24)$$

Substituting Eq. 2.24 into Eq. 2.3 leads to the desired result,

$$Q_i = \frac{-1}{u_i \rho_i L_i E_i}\left(\frac{F_{Pi}F_{Qi}L_i}{A_i{}^2{}_{,old}} + \frac{1}{(A_{i,new} - A_{i,old})}\left(\frac{(I_{yi,new} - I_{yi,old})B_{yi}}{I_{yi}{}^2{}_{,old}}\right.\right.$$

$$\left.\left. + \frac{(I_{zi,new} - I_{zi,old})B_{zi}}{I_{zi}{}^2{}_{,old}}\right)\right) \qquad (2.25)$$

2.5 Details of the Stiffness Design Process

Details of the stiffness design heuristic optimization procedure will be described in this section. All discussion will be related to the macro-flow charts of the process shown in Figs. 2.2, 2.3, and 2.4. As described in Section 2.1, the stiffness design method is specifically intended to satisfy imposed lateral deflection constraints at selected joints or to satisfy imposed rotational constraints at selected floors in a space frame.

The first step in the stiffness design procedure is to input all

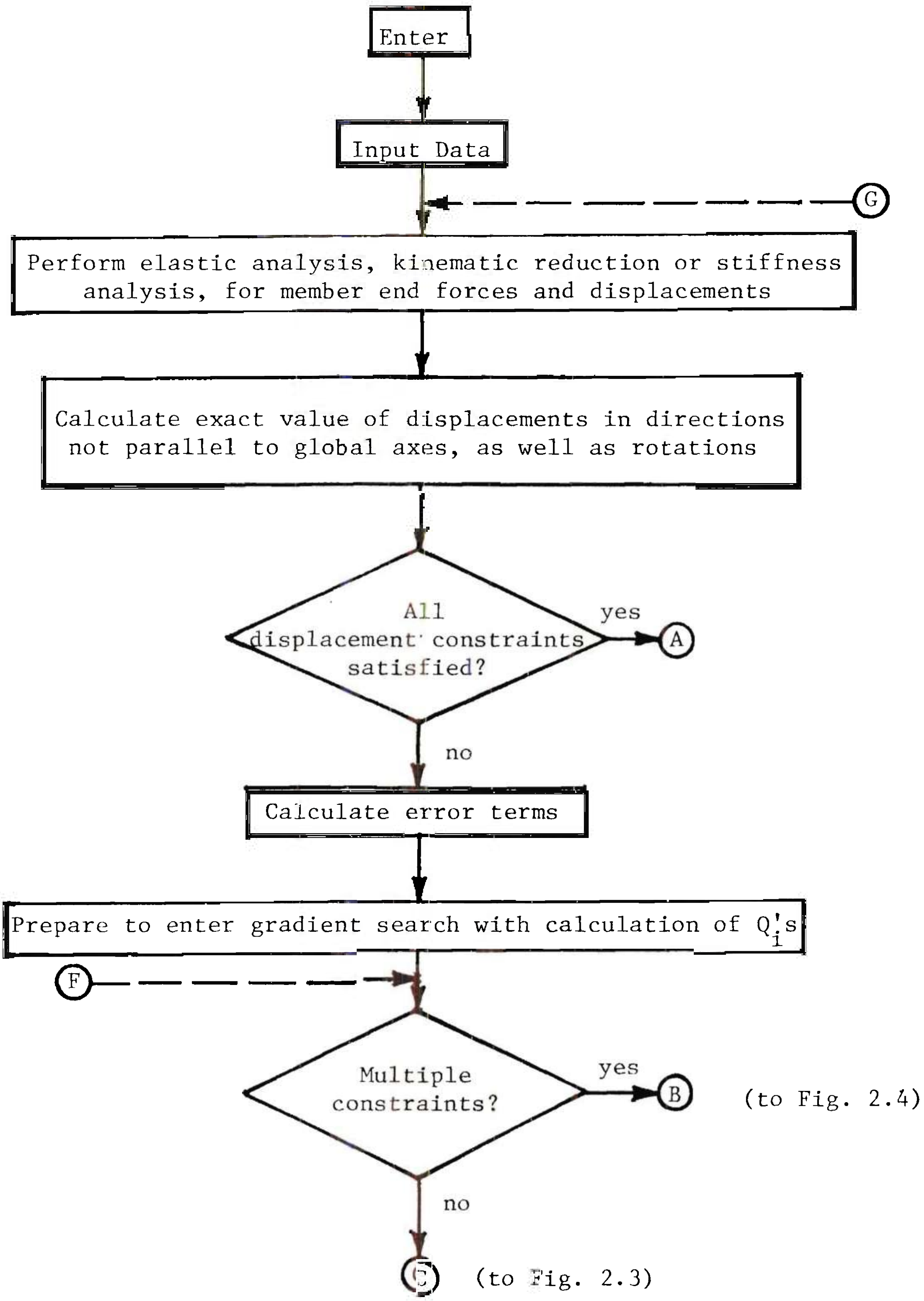


Figure 2.2 Macro Flow Chart of Overall Stiffness Design System

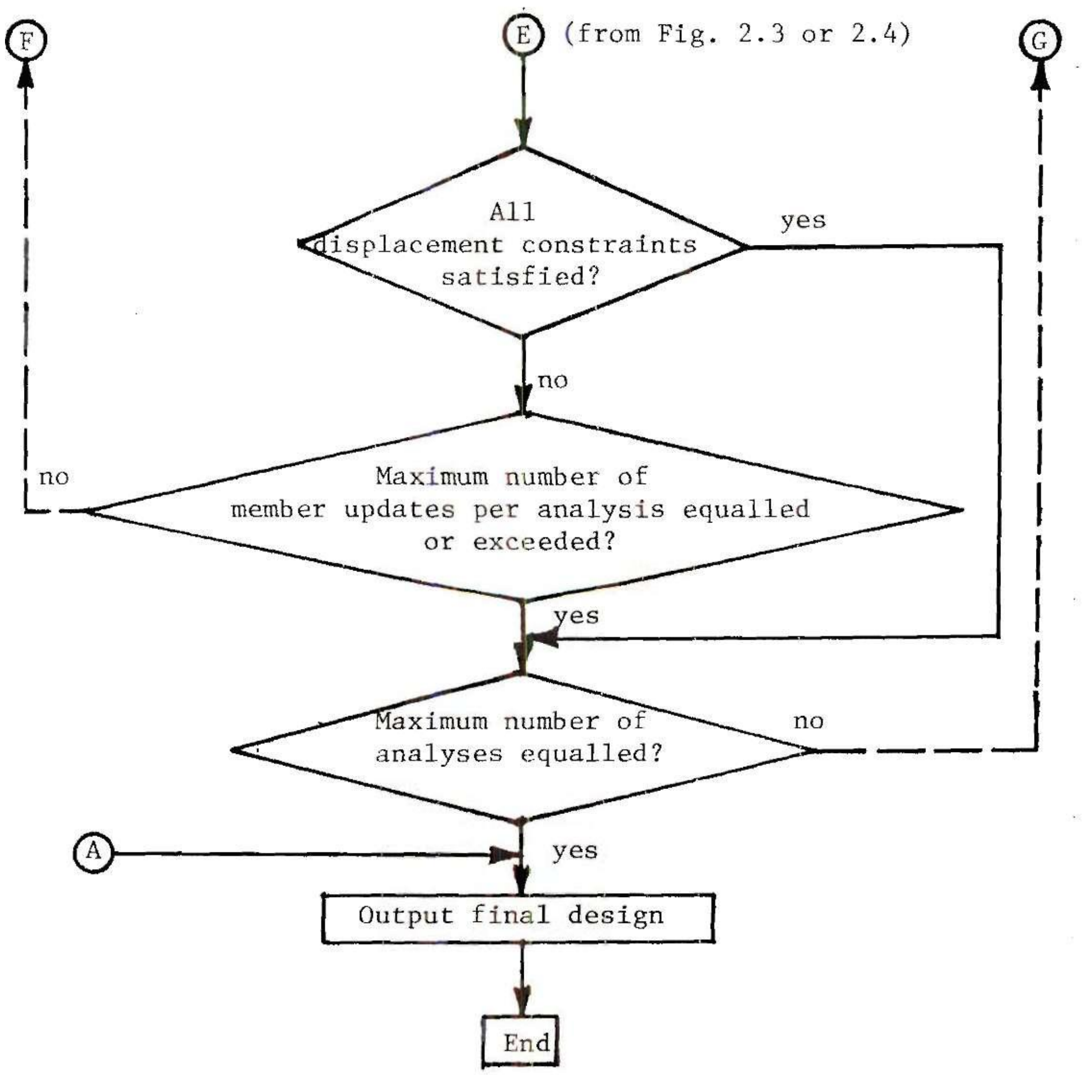


Figure 2.2 Continued

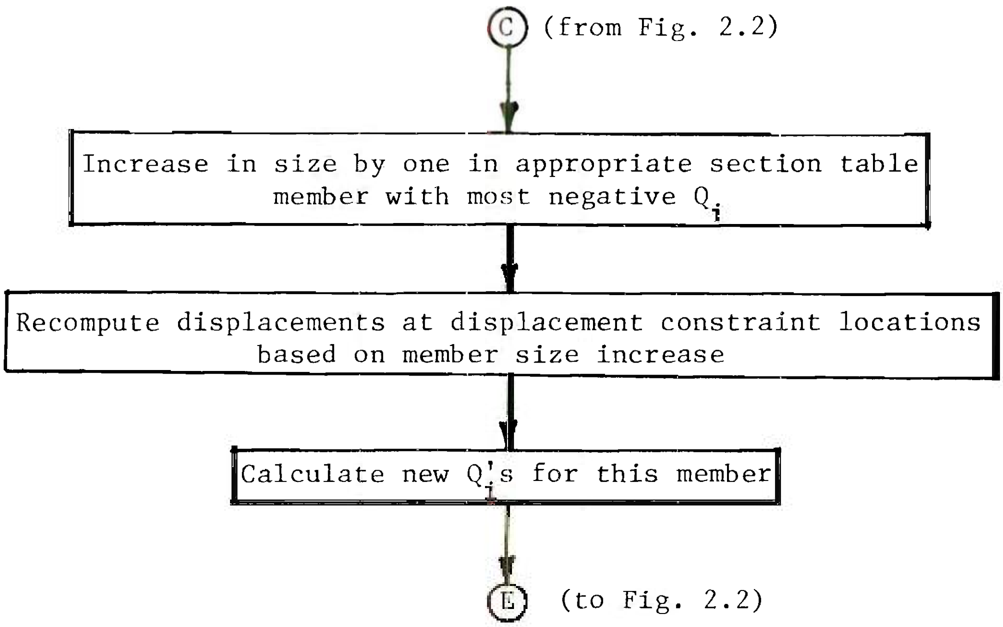


Figure 2.3 Case of Single Displacement Constraint

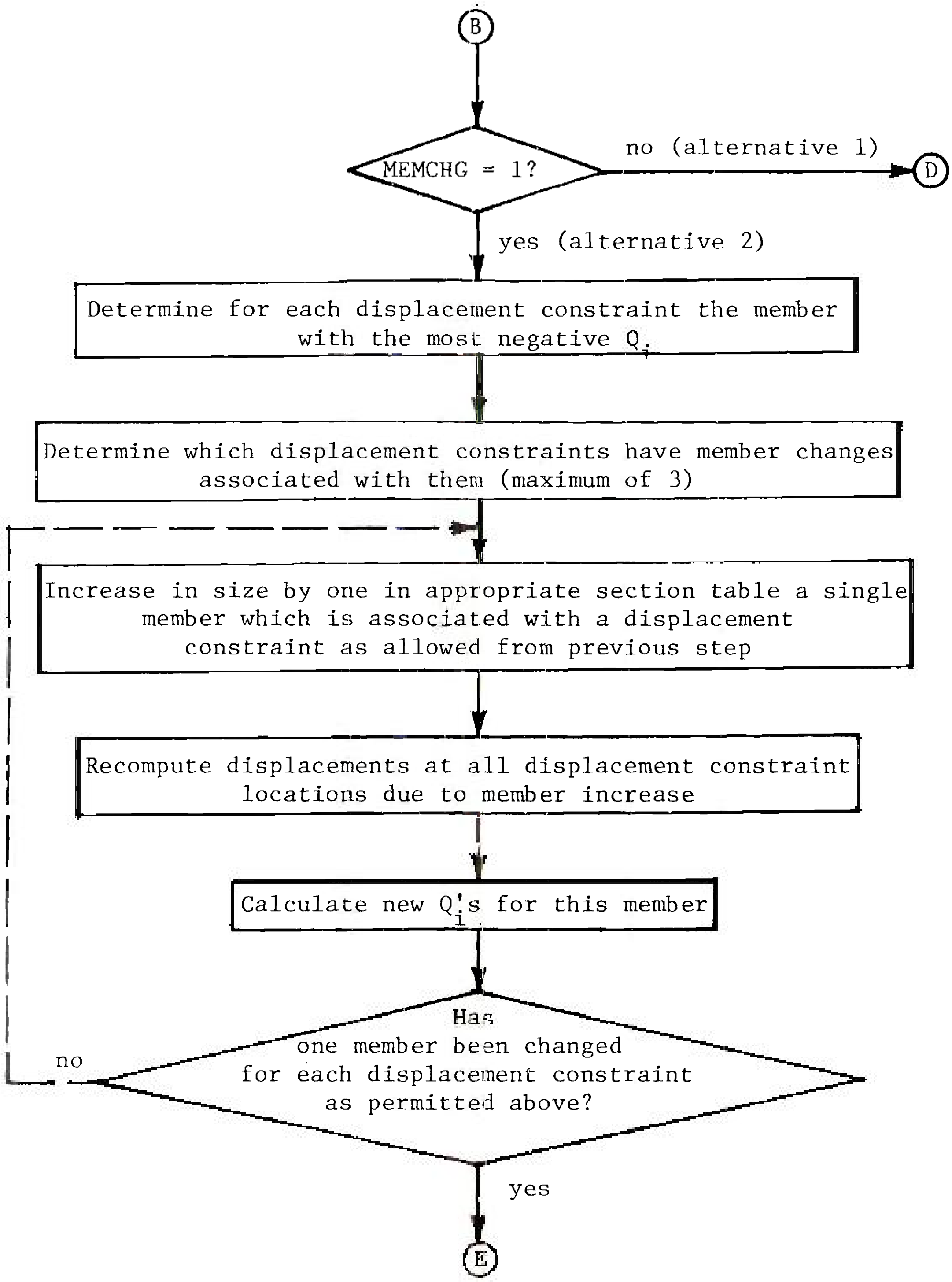


Figure 2.4 Case of Multiple Displacement Constraints

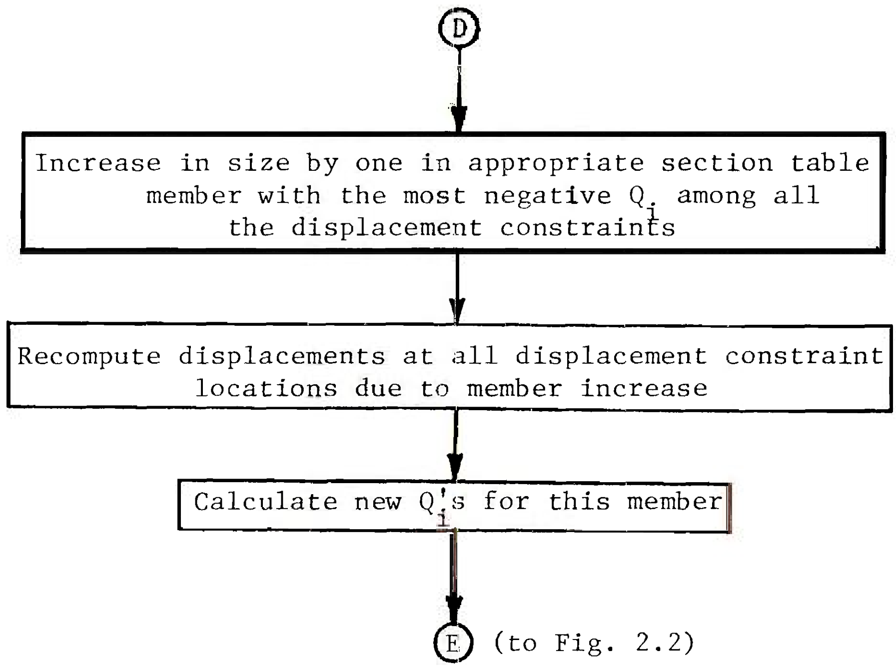


Figure 2.4 Continued

design parameters and perform consistency checks. Exact format and content of input is detailed in Appendix A. Error messages and termination of the process result when any of these parameters exceed their prescribed limits. These error messages may be found in the detailed discussion of subroutines in Appendix D.

In order to facilitate a complete understanding of how the stiffness design process works, however, certain important input parameters will be described here. First the Q-P loading table, consisting of Q load numbers and P load numbers must be input. For a concentrated translational constraint at a joint, a Q load number refers to a loading condition consisting of a unit virtual load at the joint where a constraint exists, and in the direction of that constraint. For a rotational floor constraint about an axis normal to the floor, a Q load number refers to a loading condition consisting of a unit virtual couple at a floor where a constraint exists. It should be noted that concentrated rotational constraints at a joint are not permitted at this time.

Current program limitations permit a maximum of three displacement constraints to be specified. Thus a maximum of three Q loading conditions are allowed, since each Q loading corresponds to a single displacement constraint location (i.e. joint translation or floor rotation). A P load number signifies an applied external loading condition which is determined by the user to be a possible cause of excess displacements at one or more displacement constraint points in the structure.

Now, the Q-P loading table associates Q loads and P loads. The program permits a maximum of four P loads to be associated with any single Q load. Displacements occurring at the displacement location corresponding to a Q load must meet constraint requirements under the application of each associated P load. For example, suppose the input is such that a Q load is loading number 2, and associated P loads are given as loading numbers 3 and 4. This implies that for the application of external loading conditions 3 and 4 (each taken separately), the displacement in the direction and at the location of the application of the Q unit virtual loading number 2 must satisfy the constraint value imposed by the engineer. Note that one P load may be associated with more than one Q load if desired.

As shown in Fig. 2.2 the next step following data input is to perform an analysis of the structure under consideration for all P and Q loads. This analysis may be performed in one of two ways: either by an exact linear elastic stiffness analysis, or by a kinematic condensation technique.

The exact stiffness analysis may be performed by a user provided program. The requirements for interfacing the stiffness design procedures (and programs) described herein with the user provided program are detailed in Chapter 4, where a limited capability 3-D frame analysis program is used as an example. The kinematic condensation technique is provided along with the design system as an alternative analysis method and is detailed in Chapter 3. In either case, the objective of the analysis is to compute displacements and member forces for each of the P loads, and to compute member forces for each of the Q loads, in preparation for subsequent virtual work calculations to be described later.

At this time it is necessary to note two important situations. First, the analysis program will compute displacements only in global directions, but translational constraint directions may be in directions other than global ones. Second, concerning the case of floor rotational constraints, the analysis program does not compute floor rotations, only joint rotations.

Therefore, the next step shown in Fig. 2.2 is to compute exact displacements for translational constraints not parallel to the global axes and to compute floor rotations if either one or both of these quantities are necessary. In the case of a skewed constraint direction, in order to find the corresponding exact translational displacement, the program simply combines global displacement components as specified by the user during input. In the case of determining floor rotation, the assumption is made that for rotation calculations, floors behave as rigid diaphragms. In any rigid body, each chord connecting two points in that body experiences the same rotation as any other chord within the same body. So, the user is required to specify two joints, defining a chord, on a floor which has a rotational constraint placed on it. By comparing the angle between this chord and the positive X-axis (the positive X-axis is used as a fixed reference axis), both before and after the external load is applied, and taking the difference, floor rotation is obtained.

The number of exact displacement calculations is determined by the Q-P loading table described previously. Because a Q load corresponds to a constraint location, and the displacement at this location resulting from each associated P load must be less than or equal to the constraint value, an exact displacement must be computed at this location for each associated P load. This is true for each Q load in the Q-P table.

Once all exact displacements corresponding to displacement constraint locations are computed, the next step (Fig. 2.2) is to determine whether or not these displacements satisfy allowable values within a user specified tolerance. If such is the case the program terminates and outputs final design results. On the other hand, if the displacement constraints are not satisfied the stiffness design process is begun.

Although the next step in Fig. 2.2 (i.e. the first step in the design process), is to compute error terms, a discussion of this topic is postponed to a more appropriate time when the use of these terms is considered.

Now, as shown in Fig. 2.2, the iterative stiffness design process continues with the calculation of displacement sensitivity coefficients, Q_i's, defined in Eqs. 2.3 and 2.25, for every member i in the structure. The number of Q_i's computed for each member is dependent upon the form of Q_i and the Q-P loading table.

Consider any particular Q load corresponding to a unique displacement constraint. It has one or more P loads associated with it.

Consider the virtual work expression for Q_i, Eq. 2.25. Note that all areas and moments of inertia as well as cost, density, length and modulus of elasticity factors are constant for all P loads and all Q loads for any individual member i. The factor F_{Qi}, however, is only constant for all P loads associated with any one Q load, but varies from Q load to Q load, since F_{Qi} arises from Q loads alone. The factor F_{Pi} does vary for each P load, but not for different Q loads, since F_{Pi} arises from the P loads alone. Finally the factors B_{yi} (Eq. 2.13), and B_{zi} (Eq. 2.15) vary for both P loads and Q loads since they are functions of moments

M_{Pi} and M_{Qi} which arise from P and Q loads respectively.

Consequently, since the deflection sensitivity coefficient Q_i is a function of parameters which vary for both P and Q loads, then for any member i, one Q_i coefficient must be computed for each P load associated with each of the Q loads as specified in the Q-P loading table.

There are two special circumstances which result in a fewer number of Q_i's calculated than discussed above. First it is possible that one or more displacement constraints, each of which has a Q load related to it, may already be satisfied prior to computing the Q_i's. In this situation, there are no Q_i's calculated for the P loads associated with a Q load that is related to a constraint which is satisfied. Second, there are no Q_i's calculated for any member which is found to be at the maximum section size in the table designated for that member. In this situation, all Q_i's for any such member are set equal to +10000 for the purpose of preventing such a member from being chosen as the most cost effective member to reduce a deflection. Recall from Section 2.3 that the most negative Q_i is the one selected from the gradient search procedure.

At this point in the discussion, it is assumed that only one displacement constraint has been imposed on the structure which is being designed (Fig. 2.3). The case of multiple constraints is discussed at a later time (Fig. 2.4).

Once all Q_i's are calculated, the next step (Fig. 2.3) is to increase in size, by one section in the appropriate user supplied section table, that member with the most negative Q_i. Again, the Q-P loading table controls calculations. The Q_i selected may result from any of the P loads associated with the Q load related to that constraint. The member chosen represents the least increase in cost of the frame required to decrease the deflection at the constraint location by a unit amount. Note that for the member size increase, the largest decrease in deflection will occur at the constraint location under the application of the P load for which the most negative Q_i was chosen, although decreases will also occur for the other P loads.

Following the member size increase, the next step is to recompute the displacement at the constraint location for each P load associated with the Q load related to that constraint, by determining the change in deflection caused by the member which was increased. To compute an exact change in displacement for each P load would require full scale analyses, because new force distributions within the structure are created when any member properties change. However, this would be an extremely costly process and is discarded as an alternative in favor of a virtual work solution, which assumes all member forces to remain constant, as explained in Section 2.4. It so happens that because member forces are not changing, the change in deflection at the constraint location for any P load associated with the Q load of that constraint, is identically equal to the change caused by the particular member which was increased, and is not dependent upon any other members, as shown in Eq. 2.18. So, the incremental change in deflection is computed by taking the difference between the contribution of the member to deflection before and after it is changed using the virtual work approach, Eq. 2.24. Note in Eq. 2.24 the only parameters varying are the areas and moments of inertia, relating to the new and old values.

On the first pass through this segment of the design process after an analysis has been performed, the incremental displacement decrease, contributed by the first member increase for each P load associated with the Q load of that constraint, is added to the exact displacement from the P loads at the constraint location computed earlier by the analysis process. The resulting values of displacement become the new virtual work displacements and it is these new values which are modified after ensuing member changes if they are necessary.

Fig. 2.3 shows the next step to be executed is the computation of new Q_i's for the member which was increased in size. The expression for Q_i, Eq. 2.25, shows that only the member's properties have changed due to the increase, not member forces. Because of this, the new Q_i's will be less negative than before and as a result, the same member may not be selected to change the next time since another member may now have a Q_i which is more negative than any Q_i for this member.

A point is now reached in the procedure (Fig. 2.2) where the displacement at the constraint location for each P load associated with the Q load of that constraint is compared against the constraint value to see if the displacement constraint is satisfied.

If the constraint is satisfied according to the virtual work displacement computation, the next step is to determine whether or not the maximum number of analyses allowed have been performed, and if so, the current design is output, whether or not the design is complete. If the maximum number of exact analyses allowed have not yet been performed, another exact analysis is performed to check and see if, in fact, the displacement constraint is satisfied for each P load associated with the

Q load of that constraint. If the constraint is satisfied according to the exact displacement computation, the procedure terminates with the output of the final design. If the displacement constraint is not satisfied, the stiffness design iteration begins a new cycle.

If the constraint is not satisfied according to the virtual work displacement computation, another member size change is necessary. However, a user input variable is checked. This variable specifies the maximum number of member changes permitted during stiffness design virtual work cycling before a new full scale analysis is required. If the maximum number of member changes has not been reached, the next member with the most negative Q_i is increased and the process repeated. If this number of member changes have taken place, a check is again made to see whether or not the maximum number of exact analyses have been performed. If the maximum number of exact analyses allowed have been executed, the latest design is output and the design process terminates. If not, an exact analysis is performed to check and see if, in fact, the displacement constraint is satisfied for each P load associated with the Q load of that constraint and the design process is repeated.

The process involving member size increases, followed by the recomputation of displacement corresponding to the displacement constraints by Virtual Work, and calculation of new Q_i's constitute a cycle of virtual work computations which is repeated many times (once for each member size increase). This cycle was described as it pertained to a single displacement constraint.

Now, recall that as many as 3 displacement constraints are allowed. When multiple constraints are specified, certain complex variations may occur in the cycle, although basically it remains intact. These variations are described as follows, and are shown in Fig. 2.4.

Once it is determined that more than one constraint (either 2 or 3) exists, a special user input parameter becomes significant at the next decision point in Fig. 2.4. This parameter dictates one of the following: (1) permit just a single member, associated with the most negative Q_i regardless of which constraint it arises from, to be changed in the subsequent virtual work cycle before displacements at constraint locations are checked against the constraint values; or (2) permit one member change associated with each displacement constraint to occur in the subsequent virtual work cycle before displacements at constraint locations are checked against the constraint values. In the previous discussion pertaining to a single constraint, this parameter was ineffective since only one member would be changed anyway.

For the moment, assume that alternative (1) above has been chosen by the user. Fig. 2.4 shows that in this case, the member with the most negative Q_i is increased in size, followed by displacement calculations at all constraint locations for each P load associated with each Q load as specified in the Q-P table. Although the member change was related to a single displacement constraint location, it will affect displacements at all constraint locations. Upon completion of displacement calculations, new Q_i's are computed for this member. Then a displacement check is made at all constraint locations for each P load associated with each Q load, as specified by the Q-P table, to see whether all constraints are satisfied. From this point, the stiffness design process continues in Fig. 2.2 as described before.

Now, suppose that alternative (2) above has been chosen by the user. Fig. 2.4 shows that for each displacement constraint, the member which has the most negative Q_i is determined. Although it is now permitted to change one member size for each displacement constraint, this will not necessarily occur. It may happen that one member will have the most negative Q_i relating to more than one displacement constraint direction. In that case, such a member is increased only once in the subsequent virtual work cycle, corresponding to any one of the displacement constraints for which it had the most negative Q_i.

For example, suppose for displacement constraints 1, 2, and 3, the most negative Q_i's are -4.0, -5.0, and -6.0, and result from members 7, 8, and 9 respectively. Because three different members are selected, all will be increased in the subsequent virtual work cycle. But if members 7, 8, and 7 are designated above, only two member changes will occur, namely member 7 corresponding to constraint 1, and member 8 corresponding to constraint 2. No member change will be associated with constraint 3. Yet member 7 will cause the same decrease in displacement direction 3 as would any other member with the identical Q_i. Recall that each member change is reflected by a decrease in <u>all</u> constraint directions.

The next three steps in Fig. 2.4 within the virtual work loop, are the same as before, that is, increase member size, recompute displacements by virtual work at all constraint locations, and calculate new Q_i's for the member which was changed. Following this, a check is made to see if the proper number of members have been changed. If not, the three steps are repeated for the next member corresponding to the next constraint. If the proper number of members have been changed, then a displacement check is made at all constraint locations for each P load associated with each Q load as specified by the Q-P table. As before, from this point the stiffness design process continues as shown in Fig. 2.2.

This completes the description of the flow of the stiffness design process. An explanation of the error terms, postponed earlier, will now be discussed.

By neglecting shear and torsion deformations, and assuming member forces to remain constant, one would expect a divergence between the virtual work displacements and exact displacements. It was found that a significant difference existed and that the virtual work displacements tended to decrease faster than did the exact displacements. Actual results demonstrating this fact are shown in Chapter 5. Thus when virtual work displacements were compared to constraint values, they would appear to satisfy the constraint, when in reality displacements were still excessive as indicated by a subsequent exact analysis. In order to compensate for this discrepancy, error terms were used. There was one error term for each P load associated with each Q load as specified by the Q-P table.

Now, there are two types of error terms. The first type is an initial error term and is used only in all virtual work cycles which follow the first exact analysis. It is defined as a user input percent of the value of exact displacement at constraint locations for each P load associated with each Q load as specified by the Q-P table.

Notice in Fig. 2.2 that virtual work displacements are checked at the end of each virtual work cycle. It is at these points where the error terms are added to the value of the virtual work displacements to produce pseudo increases in their absolute values, before they are compared with the constraint limits. These increases force additional member changes, which means that additional virtual work displacement decreases will occur.

The situation may occur that despite the presence of the initial error term, the virtual work displacements which appear to satisfy constraint values, actually do not satisfy them after the next exact analysis is performed to check these displacements. Recall that the first exact analysis determines whether or not any design is necessary at all.

Therefore, the second type of error term is defined as the difference between the exact displacements at constraint locations, and the virtual work displacements at constraint locations computed from the most recent virtual work cycle. The number of errors terms is determined by the Q-P table and they are applied in the same manner as the initial error terms. The only difference between the second type of error term and the initial error term is that the second type is calculated and used after the second and subsequent analyses and involves exact and virtual work displacement differences, while the initial or first type is taken as a user specified percentage of the initial exact displacement corresponding to the displacement constraint.

The purpose of the initial error term is to force a quicker convergence of the stiffness design process. This is discussed in Chapter 5.

CHAPTER 3

KINEMATIC CONDENSATION PROCEDURE

3.1 Introduction

At certain points in the process of displacement constraint design, there is a necessity to perform an "exact" analysis of the structure under consideration for the purpose of updating joint displacements and member forces. Either a rigorous exact stiffness analysis (performed by a user provided analysis program as described in Chapter 4), or a kinematic condensation process described herein may be executed, as requested by the user during input.

For displacement calculation, the kinematic condensation process was intended to be many times faster than a rigorous analysis, yet somewhat slower than the virtual work computations described in Chapter 2. It was also intended to be more accurate than the virtual work computations, although less accurate than the exact stiffness analysis. It could be referred to as a middle level displacement approximation which would provide the desired trade-off between speed and accuracy resulting in a cost efficient technique of deflection calculation. Figure 3.1 shows the relationships between the intent of the three methods of displacement calculation used in the design system.

Unfortunately, as described in Chapter 5 examples, the kinematic condensation procedure only satisfied the accuracy intent, but did not satisfy the speed intent. For the frames considered, it was, in fact, slower than Stiffness Analysis applied to the entire frame. The

Increasing Accuracy	↑	Stiffness Analysis Kinematic Condensation* Virtual Work	↓	Increasing Speed

*As shown in Chapter 5, Summary of Results, Kinematic Condensation satisfied the accuracy goal, but did not satisfy the speed goal. For the frames considered, it was, in fact, slower than Stiffness Analysis applied to the entire frame.

Fig. 3.1 Relationships Between the Intent of the Three Methods of Displacement Calculation in the Stiffness Design System

Kinematic Condensation procedure described herein is perfectly general, however, and shows promise of benefit to dynamic analysis problems. Kinematic Condensation is accomplished by permitting the engineer to make arbitrary assumptions in regard to relationships between joint displacements in a frame, where these assumptions ultimately lead to a kinematic reduction in the number of unknown displacements which need to be determined. For instance, in-place rigidity of floors, completely-rigid floors, or any other arbitrary displacement assumptions may be specified by input of appropriate parameters to the program. Details of the method follow. Examples of its use are shown in Section 5.8.

3.2 General Formulation

In the stiffness method of matrix structural analysis, the stiffness matrix $\underset{\sim}{K}$, the displacement vector $\underset{\sim}{U}$, and the load vector $\underset{\sim}{P}$ are related as follows,

$$\underset{n \times n}{\underset{\sim}{K}} \; \underset{n \times 1}{\underset{\sim}{U}} = \underset{n \times 1}{\underset{\sim}{P}} \tag{3.1}$$

where n = total number of unknown displacements.

Certain joint displacement components in the $\underset{\sim}{U}$ vector may be related to, or are dependent upon components of displacement at arbitrarily selected points. The displacement components at these arbitrarily selected points are referred to as displacement measures which form a vector $\underset{\sim}{d}$. The joint displacement components in $\underset{\sim}{U}$ which are dependent upon $\underset{\sim}{d}$ are referred to as dependent displacements. The remaining displacements in the $\underset{\sim}{U}$ vector which are not related to the displacement measures in any way are referred to as independent displacements.

Suppose there are t independent displacements and q dependent displacements in a structure under consideration. Based on this assumption, the rows and columns of the original stiffness matrix, the rows of the original displacement vector, and the rows of the original load vector may be reordered such that the independent displacements are ordered first in a submatrix $\underset{\sim}{U}_1$, followed by the dependent displacements ordered next in a submatrix $\underset{\sim}{U}_2$. The resulting reordered equations in partitioned form are,

$$\begin{array}{cccc} t\text{x}t \quad t\text{x}q & t\text{x}1 & & t\text{x}1 \\ \left[\begin{array}{c|c} \underset{\sim}{K}_{11} & \underset{\sim}{K}_{12} \\ \hline \underset{\sim}{K}_{21} & \underset{\sim}{K}_{22} \end{array}\right] & \left[\begin{array}{c} \underset{\sim}{U}_1 \\ \hline \underset{\sim}{U}_2 \end{array}\right] & = & \left[\begin{array}{c} \underset{\sim}{P}_1 \\ \hline \underset{\sim}{P}_2 \end{array}\right] \\ q\text{x}t \quad q\text{x}q & q\text{x}1 & & q\text{x}1 \end{array} \tag{3.2}$$

where q + t = n.

Further, assume each of the q displacements in $\underset{\sim}{U}_2$ is related to the r displacement measures in $\underset{\sim}{d}$ by a known transformation matrix $\underset{\sim}{T}$,

$$\underset{q\text{x}1}{\underset{\sim}{U}_2} = \underset{q\text{x}r}{\underset{\sim}{T}} \; \underset{r\text{x}1}{\underset{\sim}{d}} \tag{3.3}$$

The partitioned vector containing $\underset{\sim}{U}_1$ and $\underset{\sim}{U}_2$ in Eq. 3.2 can therefore be written as,

$$\underset{q\times 1}{\overset{t\times 1}{\left[\begin{array}{c}\mathbf{U}_1\\ \hline \mathbf{U}_2\end{array}\right]}} = \underset{q\times 1}{\overset{t\times 1}{\left[\begin{array}{c}\mathbf{U}_1\\ \hline \mathbf{Td}\end{array}\right]}} = \underset{q\times t \quad q\times r}{\overset{t\times t \quad t\times r}{\left[\begin{array}{c|c}\mathbf{I} & \mathbf{0}\\ \hline \mathbf{0} & \mathbf{T}\end{array}\right]}} \underset{r\times 1}{\overset{t\times 1}{\left[\begin{array}{c}\mathbf{U}_1\\ \hline \mathbf{d}\end{array}\right]}} \qquad (3.4)$$

Substituting Eq. 3.4 into Eq. 3.2 leads to,

$$\underset{q\times t \qquad q\times q}{\overset{t\times t \qquad t\times q}{\left[\begin{array}{c|c}\mathbf{K}_{11} & \mathbf{K}_{12}\\ \hline \mathbf{K}_{21} & \mathbf{K}_{22}\end{array}\right]}} \underset{q\times t \quad q\times r}{\overset{t\times t \quad t\times r}{\left[\begin{array}{c|c}\mathbf{I} & \mathbf{0}\\ \hline \mathbf{0} & \mathbf{T}\end{array}\right]}} \underset{r\times 1}{\overset{t\times 1}{\left[\begin{array}{c}\mathbf{U}_1\\ \hline \mathbf{d}\end{array}\right]}} = \underset{q\times 1}{\overset{t\times 1}{\left[\begin{array}{c}\mathbf{P}_1\\ \hline \mathbf{P}_2\end{array}\right]}} \qquad (3.5)$$

Performing the indicated operations results in,

$$\underset{q\times t \qquad q\times r}{\overset{t\times t \qquad t\times r}{\left[\begin{array}{c|c}\mathbf{K}_{11} & \mathbf{K}_{12}\mathbf{T}\\ \hline \mathbf{K}_{21} & \mathbf{K}_{22}\mathbf{T}\end{array}\right]}} \underset{r\times 1}{\overset{t\times 1}{\left[\begin{array}{c}\mathbf{U}_1\\ \hline \mathbf{d}\end{array}\right]}} = \underset{q\times 1}{\overset{t\times 1}{\left[\begin{array}{c}\mathbf{P}_1\\ \hline \mathbf{P}_2\end{array}\right]}} \qquad (3.6)$$

It should be noted that Eq. 3.6 has $t + q$ equilibrium equations which correspond to the $t + q$ load components in $\mathbf{P}$. However, these equations do not all correspond to the $t + r$ displacement components in the new $\mathbf{U}$ vector. In particular, the r displacement measures do not have a one to one correspondence to the last q equilibrium equations nor the last q load components which are in $\mathbf{P}_2$. In other words, the new $\mathbf{K}$ matrix is not even a square matrix (it is a $(q+t)\mathrm{x}(t+r)$). To remedy this situation, Eq. 3.6 is premultiplied by,

$$\begin{array}{c} \text{txt} \quad \text{txq} \\ \left[\begin{array}{c|c} \underset{\sim}{I} & \underset{\sim}{O} \\ \hline \underset{\sim}{O} & \underset{\sim}{T}^t \end{array}\right] \\ \text{rxt} \quad \text{rxq} \end{array}$$

and this leads to,

$$\begin{array}{c} \text{txt} \quad \text{txq} \\ \left[\begin{array}{c|c} \underset{\sim}{I} & \underset{\sim}{O} \\ \hline \underset{\sim}{O} & \underset{\sim}{T}^t \end{array}\right] \\ \text{rxt} \quad \text{rxq} \end{array} \begin{array}{c} \text{txt} \quad \text{txr} \\ \left[\begin{array}{c|c} \underset{\sim}{K}_{11} & \underset{\sim}{K}_{12}\underset{\sim}{T} \\ \hline \underset{\sim}{K}_{21} & \underset{\sim}{K}_{22}\underset{\sim}{T} \end{array}\right] \\ \text{qxt} \quad \text{qxr} \end{array} \begin{array}{c} \text{txl} \\ \left[\begin{array}{c} \underset{\sim}{U}_1 \\ \hline \underset{\sim}{d} \end{array}\right] \\ \text{rxl} \end{array} = \begin{array}{c} \text{txt} \quad \text{txq} \\ \left[\begin{array}{c|c} \underset{\sim}{I} & \underset{\sim}{O} \\ \hline \underset{\sim}{O} & \underset{\sim}{T}^t \end{array}\right] \\ \text{rxt} \quad \text{rxq} \end{array} \begin{array}{c} \text{txl} \\ \left[\begin{array}{c} \underset{\sim}{P}_1 \\ \hline \underset{\sim}{P}_2 \end{array}\right] \\ \text{qxl} \end{array} \tag{3.7}$$

The result after performing the indicated matrix operations is (t + r) kinematically reduced equations in terms of (t + r) unknown displacements,

$$\begin{array}{c} \text{txt} \quad \text{txr} \\ \left[\begin{array}{c|c} \underset{\sim}{K}_{11} & \underset{\sim}{K}_{12}\underset{\sim}{T} \\ \hline \underset{\sim}{T}^t\underset{\sim}{K}_{21} & \underset{\sim}{T}^t\underset{\sim}{K}_{22}\underset{\sim}{T} \end{array}\right] \\ \text{rxt} \quad \text{rxr} \end{array} \begin{array}{c} \text{txl} \\ \left[\begin{array}{c} \underset{\sim}{U}_1 \\ \hline \underset{\sim}{d} \end{array}\right] \\ \text{rxl} \end{array} = \begin{array}{c} \text{txl} \\ \left[\begin{array}{c} \underset{\sim}{P}_1 \\ \hline \underset{\sim}{T}^t\underset{\sim}{P}_2 \end{array}\right] \\ \text{rxl} \end{array} \tag{3.8}$$

Recall from Eq. 3.3 that, $\underset{\sim}{U}_2 = \underset{\sim}{T}\,\underset{\sim}{d}$.

The resulting kinematically reduced stiffness matrix in Eq. 3.8

is therefore symmetric. Note that the total number of unknown displacements has been reduced from n, in Eq. 3.1, to t + r, in Eq. 3.8, using this kinematic condensation procedure. Eq. 3.8 may now be solved for the t independent joint displacements in $\underset{\sim}{U}_1$, the r displacement measures in $\underset{\sim}{d}$, and the q dependent joint displacements in $\underset{\sim}{U}_2$ by Eq. 3.3.

3.3 Assembling the Transformation Matrix, $\underset{\sim}{T}$

From Eq. 3.3, it is seen that the transformation matrix $\underset{\sim}{T}$ is one which relates the r displacement measures, $\underset{\sim}{d}$, to the q dependent joint displacements, $\underset{\sim}{U}_2$. The exact form of $\underset{\sim}{T}$ therefore depends upon the desired relationship between displacement measures $\underset{\sim}{d}$ and dependent joint displacements $\underset{\sim}{U}_2$. Two examples of such a relationship are shown in what follows.

3.3.1 $\underset{\sim}{T}$ for In-Plane Floor Rigidity

Suppose it is desired to have each floor k, assumed to lie in the horizontal X-Z plane of a large building frame, simulate a rigid diaphragm in its own plane. Consequently, each joint on the floor is constrained to displace in a pattern dictated by rigid body floor motion. The in-plane translations, X and Z, and planar rotation about a vertical axis, Y, at each joint may therefore be related to the in-plane displacements at an arbitrarily selected point P_k in the floor k rigid body. Planar displacements at point P_k shown in Figure 3.2 as d_{kx}, d_{kz} and r_{ky} are the displacement measures, while displacements at the joints in the floor consist of both independent and dependent displacements. In Fig. 3.2(a), an arbitrary joint j in floor k has degrees of freedom $u_{j,t}$ where j is the joint number, and t is the local direction of displacement as shown in Figure 3.2(b).

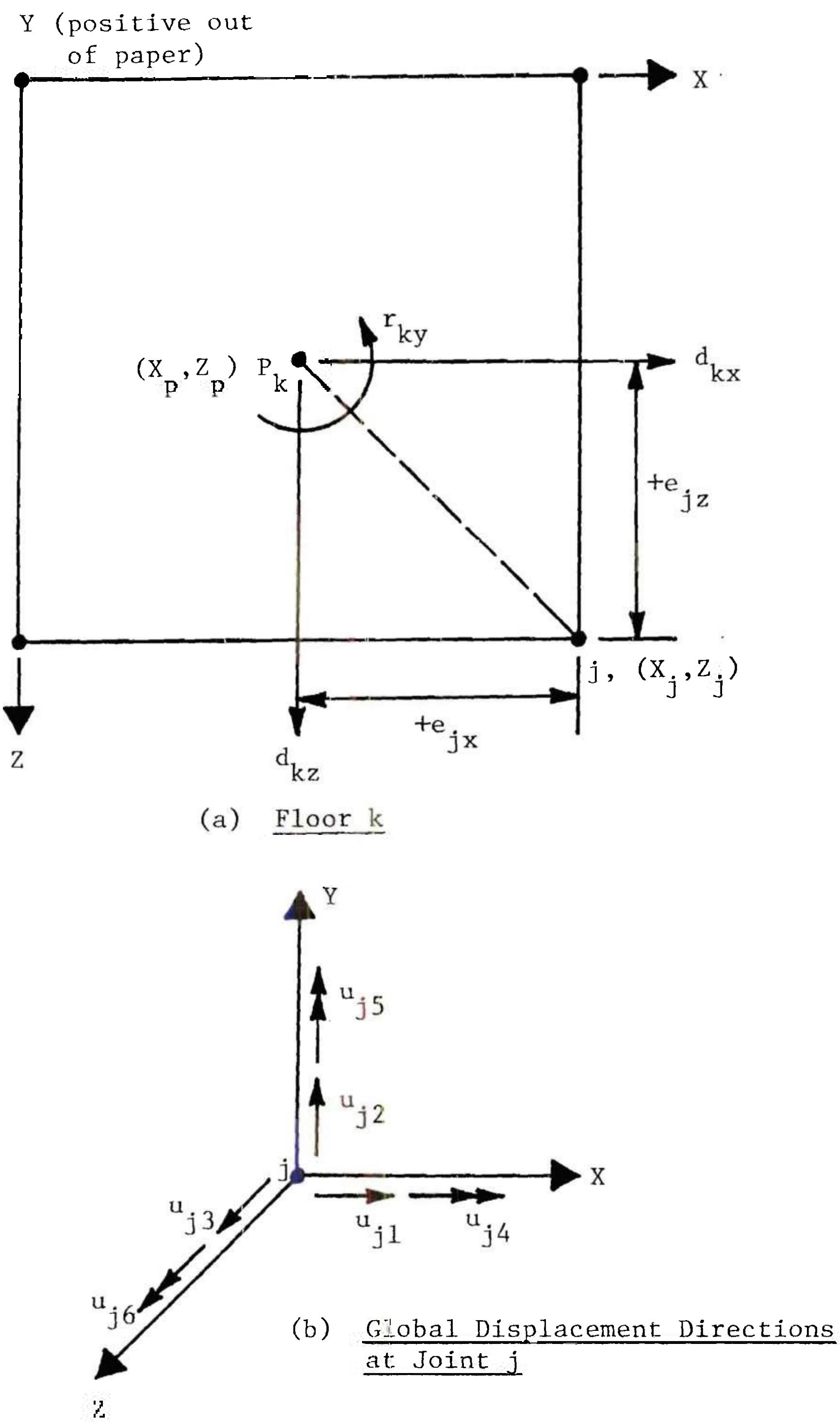


Figure 3.2 Floor k - In-Plane Floor Rigidity

Now, for a rigid floor diaphragm, u_{j1}, u_{j3}, and u_{j5} are dependent displacements, while u_{j2}, u_{j4}, and u_{j6} are independent displacements. The rigid body equations of motion describe the relationships between the dependent displacements and the displacement measures. For joint j they are,

$$u_{j1} = d_{kx} + e_{jz} * r_{ky}$$

$$u_{j3} = d_{kz} - e_{jx} * r_{ky}$$

$$u_{j5} = r_{ky} \tag{3.9}$$

where,

$$e_{jx} = X_j - X_p$$

$$e_{jz} = Z_j - Z_p \tag{3.10}$$

and where (X_j, Z_j) and (X_p, Z_p) are the X and Z coordinates of joint j and point P respectively.

In matrix form, Eq. 3.9 can be written as,

$$\begin{bmatrix} u_{j1} \\ u_{j3} \\ u_{j5} \end{bmatrix} = \begin{bmatrix} 1 & 0 & e_{jz} \\ 0 & 1 & -e_{jx} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} d_{kx} \\ d_{kz} \\ r_{ky} \end{bmatrix} \tag{3.11}$$

Now let,

$$\underset{\sim}{U}_2^j = \begin{bmatrix} u_{j1} \\ u_{j3} \\ u_{j5} \end{bmatrix}, \quad \underset{\sim}{U}_1^j = \begin{bmatrix} u_{j2} \\ u_{j4} \\ u_{j6} \end{bmatrix}$$

$$\underset{\sim}{T}_j = \begin{bmatrix} 1 & 0 & e_{jz} \\ 0 & 1 & -e_{jx} \\ 0 & 0 & 1 \end{bmatrix}$$

$$\underset{\sim}{d}_k = \begin{bmatrix} d_{kx} \\ d_{kz} \\ r_{ky} \end{bmatrix} \tag{3.12}$$

So, for joint j in floor k, using Eq. 3.12 notation, Eq. 3.11 can be written as,

$$\underset{\sim}{U}_2^j = \underset{\sim}{T}_j \underset{\sim}{d}_k \tag{3.13}$$

Now, the $\underset{\sim}{T}_j$ for each joint j, in each floor k, of a multistory building having rigid floor diaphragms must now be assembled into $\underset{\sim}{T}$ for the entire structure. An example will be used to show how this is accomplished.

Consider the example in Fig. 3.3, which shows a 2 story, 1 bay by 1 bay space frame. In this structure, it is assumed that all joints

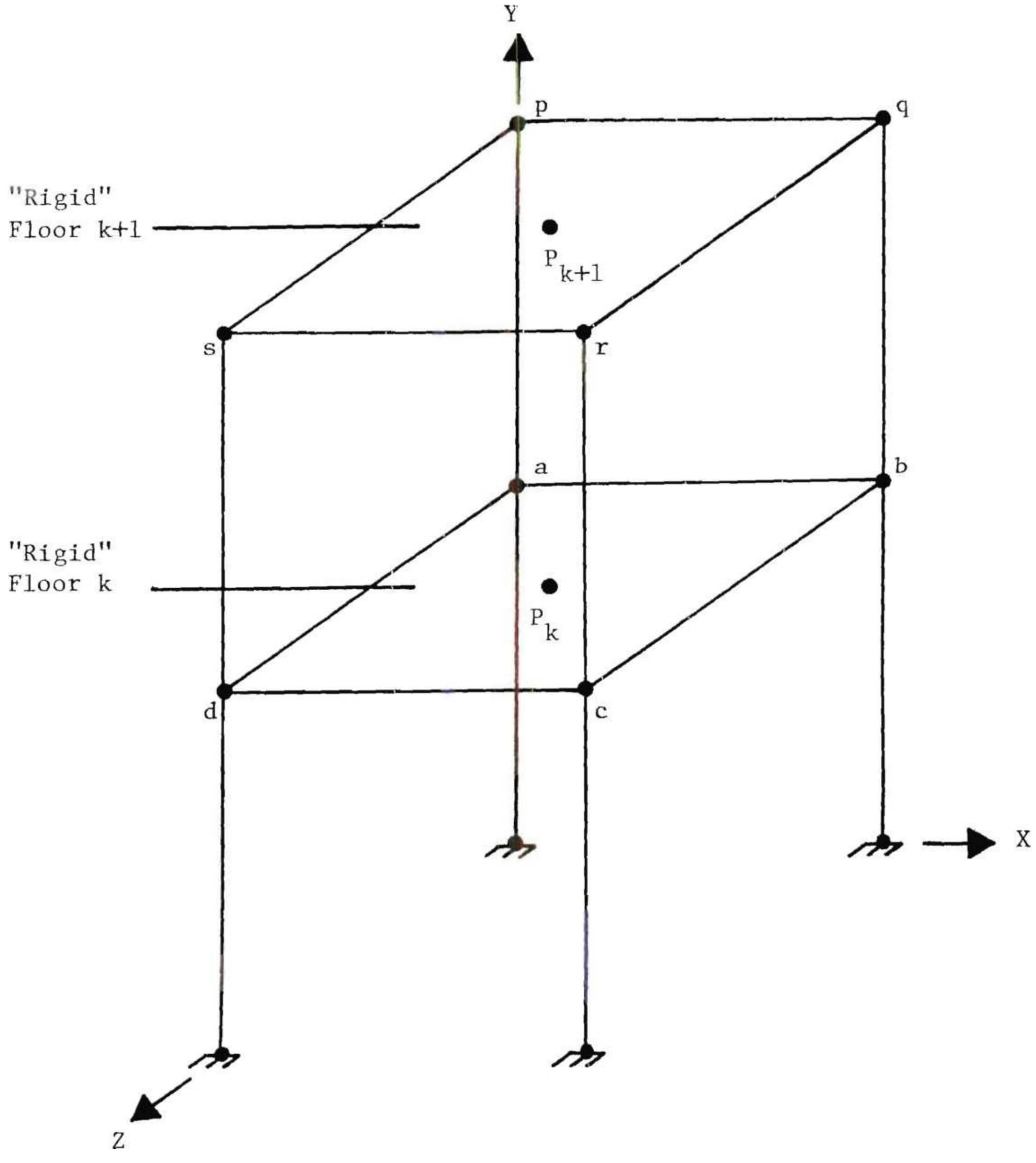


Figure 3.3 Two Story, One Bay x One Bay Space Frame

with dependent displacements relating to the same displacement measures are numbered consecutively. The significance of consecutive numbering of these joints in a story will be seen in what follows. Joints a,b,c, and d on floor k are associated with point P_k, while joints p,q,r, and s in floor k + 1 are associated with point P_{k+1}. Floors are numbered from the bottom up in this case. Knowing this information, the $\underset{\sim}{U}_2$, $\underset{\sim}{T}$, and $\underset{\sim}{d}$ matrices for this structure are assembled as follows,

$$\begin{bmatrix} \underset{\sim}{U}_2^a \\ \underset{\sim}{U}_2^b \\ \underset{\sim}{U}_2^c \\ \underset{\sim}{U}_2^d \\ \underset{\sim}{U}_2^p \\ \underset{\sim}{U}_2^q \\ \underset{\sim}{U}_2^r \\ \underset{\sim}{U}_2^s \end{bmatrix} = \left[\begin{array}{c|c} \underset{\sim}{T}_a & \underset{\sim}{0} \\ \underset{\sim}{T}_b & \underset{\sim}{0} \\ \underset{\sim}{T}_c & \underset{\sim}{0} \\ \underset{\sim}{T}_d & \underset{\sim}{0} \\ \underset{\sim}{0} & \underset{\sim}{T}_p \\ \underset{\sim}{0} & \underset{\sim}{T}_q \\ \underset{\sim}{0} & \underset{\sim}{T}_r \\ \underset{\sim}{0} & \underset{\sim}{T}_s \end{array}\right] \begin{bmatrix} \underset{\sim}{d}_k \\ \underset{\sim}{d}_{k+1} \end{bmatrix} \tag{3.14}$$

and where,

$$\underset{\sim}{U}_2 = \begin{bmatrix} \underset{\sim}{U}_2^a \\ \underset{\sim}{U}_2^b \\ \underset{\sim}{U}_2^c \\ \underset{\sim}{U}_2^d \\ \underset{\sim}{U}_2^p \\ \underset{\sim}{U}_2^q \\ \underset{\sim}{U}_2^r \\ \underset{\sim}{U}_2^s \end{bmatrix}, \quad \underset{\sim}{U}_1 = \begin{bmatrix} \underset{\sim}{U}_1^a \\ \underset{\sim}{U}_1^b \\ \underset{\sim}{U}_1^c \\ \underset{\sim}{U}_1^d \\ \underset{\sim}{U}_1^p \\ \underset{\sim}{U}_1^q \\ \underset{\sim}{U}_1^r \\ \underset{\sim}{U}_1^s \end{bmatrix} \tag{3.15}$$

$$\underset{\sim}{T} = \left[\begin{array}{c|c} \underset{\sim}{T}_a & \underset{\sim}{0} \\ \underset{\sim}{T}_b & \underset{\sim}{0} \\ \underset{\sim}{T}_c & \underset{\sim}{0} \\ \underset{\sim}{T}_d & \underset{\sim}{0} \\ \underset{\sim}{0} & \underset{\sim}{T}_p \\ \underset{\sim}{0} & \underset{\sim}{T}_q \\ \underset{\sim}{0} & \underset{\sim}{T}_r \\ \underset{\sim}{0} & \underset{\sim}{T}_s \end{array}\right] \tag{3.16}$$

$$\underset{\sim}{d} = \begin{bmatrix} \underset{\sim}{d}_k \\ \underset{\sim}{d}_{k+1} \end{bmatrix} \tag{3.17}$$

where $\underset{\sim}{U}_1^j$, $\underset{\sim}{U}_2^j$, $\underset{\sim}{T}_j$, and $\underset{\sim}{d}_k$ are given by Eq. 3.12.

Note that each partitioned column of $\underset{\sim}{T}$ corresponds to a floor. The first column of $\underset{\sim}{T}$ in Eq. 3.16 shows the transformation of joints on floor k, and the second column does the same for floor k + 1. Because joints are numbered consecutively, the $\underset{\sim}{T}_j$ and $\underset{\sim}{0}$ submatrices cluster together as shown. Also, should there be more joints on a floor than four as illustrated in this example, there would be additional $\underset{\sim}{T}_j$ submatrices in the partitioned column of $\underset{\sim}{T}$ corresponding to the floor where each of these joints is located. Also, the $\underset{\sim}{U}_2$ vector would be expanded to include the displacements from the new joints. If the structure consisted of more than two floors, an additional partitioned column of $\underset{\sim}{T}$ and a new entry in $\underset{\sim}{d}$ would be required for each extra floor, together with new contributions to the $\underset{\sim}{U}_2$ vector corresponding to the joints on the added floors. It should be noted that for the case of in-plane floor rigidity, it is not required that a $\underset{\sim}{T}$ matrix be input, as this condition is programmed internally, and this alternative may be selected by the user. Also, although consecutive joint numbers in a floor are significant in the form of $\underset{\sim}{T}$, it is not necessary that such a format be followed for proper execution of the kinematic condensation procedure.

3.3.2 $\underset{\sim}{T}$ for Complete Floor Rigidity

Another case of interest is that of complete floor rigidity, both in and out of plane. All displacements at each joint in any given floor k are dependent displacements, and are related to the six displacement measures at an arbitrarily selected point P_k in the floor k rigid body. Figure 3.4 shows point P_k and its displacements, each of which is a rigid body floor displacement measure. Joint j has deflections $u_{j,t}$

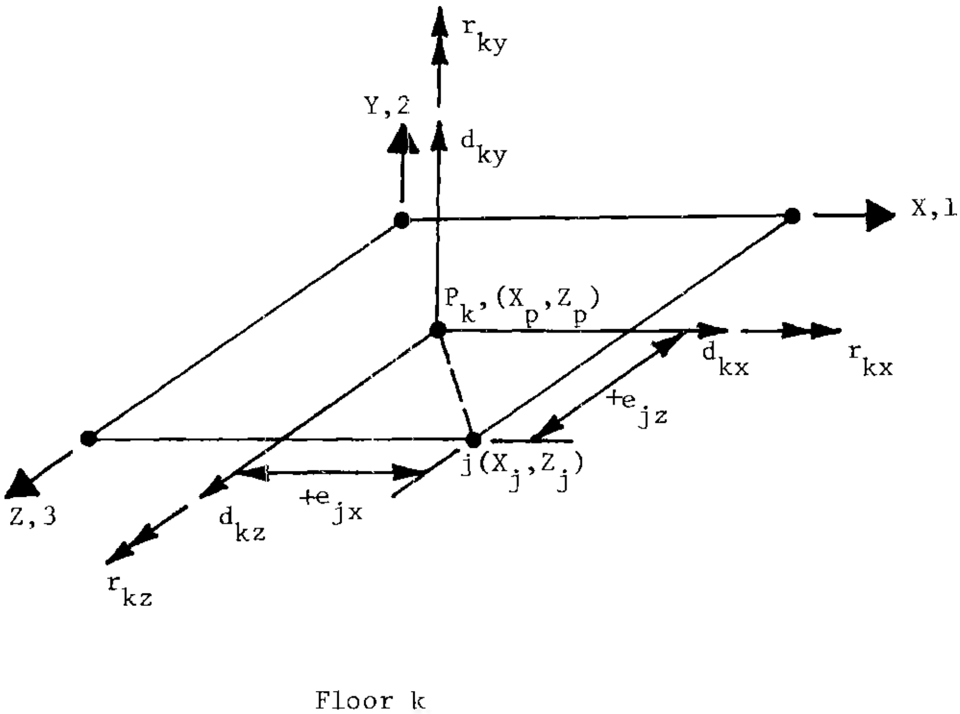


Figure 3.4 Floor k - Complete Floor Rigidity

defined as before and shown in Fig. 3.2(b), all of which are dependent on the floor k displacement measures. There are no independent displacements at joint j in this example.

Once again, the rigid body equations of motion are employed to describe the relationships between the joint j dependent displacements and the floor k displacement measures. These are,

$$
\begin{aligned}
u_{j1} &= d_{kx} + r_{ky} * e_{jz} \\
u_{j2} &= d_{ky} - r_{kx} * e_{jz} + r_{kz} * e_{jx} \\
u_{j3} &= d_{kz} - r_{ky} * e_{jx} \\
u_{j4} &= r_{kx} \\
u_{j5} &= r_{ky} \\
u_{j6} &= r_{kz}
\end{aligned} \tag{3.18}
$$

where, e_{jx} and e_{jz} are defined in Eq. 3.9. In matrix form, Eq. 3.18 can be written as,

$$
\begin{bmatrix} u_{j1} \\ u_{j2} \\ u_{j3} \\ u_{j4} \\ u_{j5} \\ u_{j6} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & e_{jz} & 0 \\ 0 & 1 & 0 & -e_{jz} & 0 & e_{jx} \\ 0 & 0 & 1 & 0 & -e_{jx} & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} d_{kx} \\ d_{ky} \\ d_{kz} \\ r_{kx} \\ r_{ky} \\ r_{kz} \end{bmatrix} \tag{3.19}
$$

Now, let

$$\underset{\sim}{U}_2^j \quad \begin{bmatrix} u_{j1} \\ u_{j2} \\ u_{j3} \\ u_{j4} \\ u_{j5} \\ u_{j6} \end{bmatrix} \quad , \text{ No } \underset{\sim}{U}_1^j$$

$$\underset{\sim}{T}_j = \begin{bmatrix} 1 & 0 & 0 & 0 & e_{jz} & 0 \\ 0 & 1 & 0 & -e_{jz} & 0 & e_{jx} \\ 0 & 0 & 1 & 0 & -e_{jx} & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\underset{\sim}{d}_k \quad \begin{bmatrix} d_{kx} \\ d_{ky} \\ d_{kz} \\ r_{kx} \\ r_{ky} \\ r_{kz} \end{bmatrix} \qquad (3.20)$$

It should be noted that $\underset{\sim}{U}_2^j$ contains all the displacement components at joint j, since no independent displacements exist, and therefore, $\underset{\sim}{U}_1^j$ does not exist.

Consider again the structure shown in Fig. 3.3. Given that this structure is now to be modeled having perfectly rigid floors, the $\underset{\sim}{U}_2$, $\underset{\sim}{T}$, and $\underset{\sim}{d}$ matrices for the structure are the same form as Eqs. 3.14 to 3.17, except that the submatrices $\underset{\sim}{U}_2^j$, $\underset{\sim}{T}_j$ and $\underset{\sim}{d}_k$ are now defined by Eq. 3.20. It should be noted that for the case of total floor rigidity, it is not required that a $\underset{\sim}{T}$ matrix be input, as this condition is programmed internally, and this alternative may be selected by the user. Also, although consecutive joint numbers in a floor are significant in the form of $\underset{\sim}{T}$, it is not necessary that such a format be followed for proper execution of the kinematic condensation procedure.

3.3.3 Effect of Kinematic Condensation on the Reduced Equations

Eqs. 3.1 and 3.8 show the total number of unknowns are reduced from n to t + r by the kinematic condensation process. This is considered in more detail in this Section.

The initial number of unknown displacements, n, in a space frame structure is,

$$n = (6 * NJ) - NR \tag{3.21}$$

where,

NJ = number of joints in the structure

NR = number of reactions in the structure

For the case of rigid in-plane floor diaphragms, with 3 displacement measures per floor, the number of unknown displacements, $NROW_3$, becomes,

$$NROW_3 = \sum_{k=1}^{NF} \left[(3 * JF_k) + 3\right] \tag{3.22}$$

where,

NF = number of floors in the structure

JF_k = number of joints on floor k

The term $(3 * JF_k)$ represents the 3 independent out of plane deflections at each joint in floor k, while the + 3 indicates the 3 independent displacement measures for floor k. When summed up over all floors, the total number of unknown structure displacements is found.

For the case of complete floor rigidity, with 6 displacement measures per floor, the number of unknown displacement components, $NROW_6$, is

$$NROW_6 = 6 * NF \tag{3.23}$$

It should be observed that Eqs. 3.22 and 3.23 presume that each joint on each floor is part of the floor rigid bodies (i.e. although not a requirement of the design system, for purposes of this discussion, there are no joints located between floors).

Now, Section 3.1 states that the kinematic condensation procedure is intended to reduce the number of actual displacements to be calculated in a given structure. Using the structure in Fig. 3.3 as an example, n, $NROW_3$, and $NROW_6$ are computed below.

$$n = (6 * 12) - 24$$
$$= 48$$

$$NROW_3 = \sum_{k=1}^{2} \left[(3 * JF_k) + 3\right] = \left[(3 * 4) + 3\right] + \left[(3 * 4) + 3\right]$$
$$= 30$$

$$NROW_6 = 6 * 2$$
$$= 12$$

Thus, the assumption of in-plane floor rigidity reduces the total unknown displacements from 48 to 30. Similarly, a reduction from 48 to 12 unknown displacements is realized when full floor rigidity is assumed.

Various other types of behavior and joint displacement relationships may be assumed by the user, by simply inputing the proper $\tilde{T}$ matrix. This $\tilde{T}$ matrix is formulated in a similar manner to the $\tilde{T}$ matrices developed in Sections 3.3.1 and 3.3.2. However, the engineer must decide which type of special behavior, if any, is valid for the building frame in question. The cases of in-plane rigidity and full floor rigidity are internally programmed into the system, since many tier building frames very closely approximate one of these assumptions in their actual displacement patterns.

Consequently, several alternatives are available to the engineer user. First he may assemble and input the $\tilde{T}$ matrix as part of the data. Second, he may request the system to use the in-plane or full floor rigidity condition to automatically assemble $\tilde{T}$. Finally, he may decide not to use a transformation matrix at all. Necessary input formats for each situation are described in detail in Appendix A.

Now presumably, reducing the number of unknowns results in a system of equations which require less computer time to solve. However, the time to solve is also a function of the band width. Consequently, it is important to note the effect of kinematic condensation on the band width.

To illustrate, consider the final reordered stiffness matrix, Eq. 3.2, and associated with in-plane floor rigidity. The first t rows of this matrix correspond to t independent displacement directions at joints within the structure. For buildings where all joints are located in the floors (the usual case), each joint has one or more dependent displacements and each joint corresponding to an independent displacement is connected to another joint which contains one or more dependent displacements. Therefore upon reordering, the submatrix $\tilde{K}_{12}$, which is defined by the intersection of the t independent rows and q dependent columns of the reordered matrix, will have some nonzero entries in every row.

Now, consider the final kinematically condensed stiffness matrix from Eq. 3.8. Notice that $\tilde{K}_{12}$ is post multiplied by $\tilde{T}$. The form of $\tilde{T}$, regardless of which points are located on each floor (where each floor is a rigid body with all joints on it containing dependent displacements), will have some nonzero entries in every column. Therefore, upon conclusion of the matrix operation $\tilde{K}_{12}\tilde{T}$, a full kinematically reduced stiffness matrix exists, meaning that the band width and order of the matrix are identical. So, although there are fewer unknowns, the time required to solve may increase due to a larger band width.

Solving time is directly related to the size of the matrix in question and approximately the square of its band width. Consider the following example. Suppose the structure in Fig. 3.3 is numbered such that the connectivity for members results in the largest difference between any member incidences of 4, and because there are 6 unknowns per joint, the band width becomes 24. With 8 joints completely free to displace, there are 48 unknown displacements. This same structure when modeled with in-plane rigid floors has 30 unknown displacements. From the discussion above, the band width will also be 30. So, the comparison is made as follows,

$$(48)(24)^2 = 27648$$

$$(30)(30)^2 = 27000$$

In this case, apparent solution times between the two situations would be approximately equal, if the time necessary to reduce the stiffness matrix is neglected. But, the reduction time was found to be very high in comparison with solution time and thus the kinematic condensation procedure is slower than an exact analysis.

In more general terms, and neglecting reduction time, the following comparison of solution times can be made,

$$nb^2 \geq n_r(n_r)^2 \tag{3.24}$$

where,

n = total number of unknown displacements

b = band width of original stiffness matrix

n_r = number of unknown displacements in kinematically condensed system

This implies that kinematic condensation would be suitable if,

$$b \geq n_r \sqrt{\frac{n_r}{n}} \tag{3.25}$$

However, for most tier buildings, including the examples discussed in Chapter 5, this band width criteria is not met and thus the kinematic condensation is more costly than an exact analysis. But, as will be shown in Section 5.8, the results obtained from this procedure are accurate if the proper assumptions are made. It is very important to note that the inefficiency of the procedure was not recognized at the beginning of the investigation. It was not until the entire procedure was developed and implemented that this was discovered. Consequently, although the kinematic condensation is perfectly general, it is not recommended for use in this stiffness design method.

3.4 Details of the Kinematic Condensation Procedure

The details of the kinematic condensation procedure will be presented in this section. All discussion will be related to the macro flow chart of the procedure shown in Fig. 3.5.

As described in Section 3.1, the kinematic condensation procedure is specifically intended to serve as a middle level displacement approximation, between exact analysis and virtual work computations of displacements. The final kinematically condensed stiffness equations, Eq. 3.8, can be formed by straightforward matrix operations. However, the sizes of the matrices involved, $\underset{\sim}{T}$ and $\underset{\sim}{K}$, as compared to the core storage capacity of most computers, severely limits the structure size

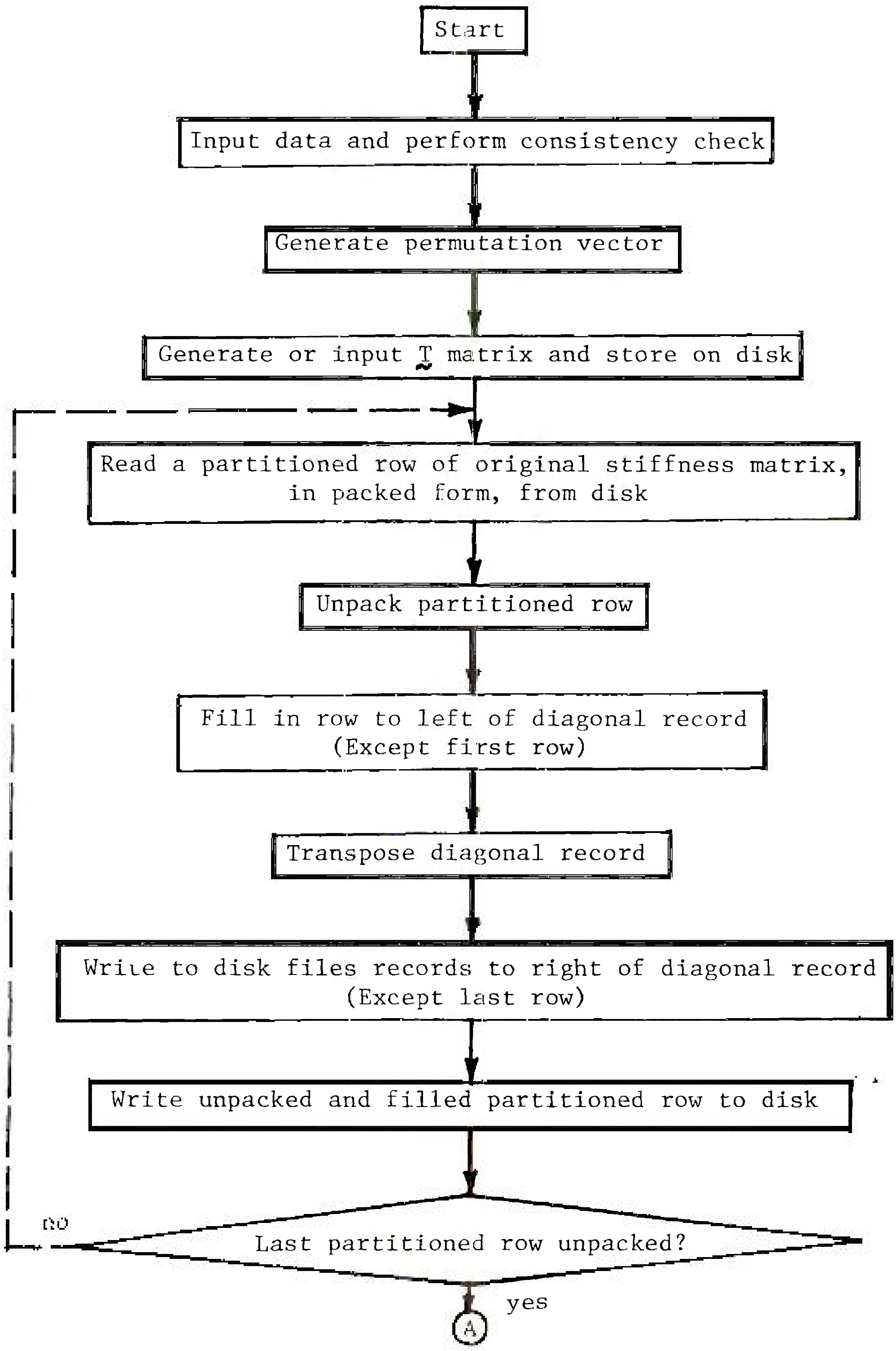


Figure 3.5 Macro Flow Chart of Kinematic Condensation Procedure

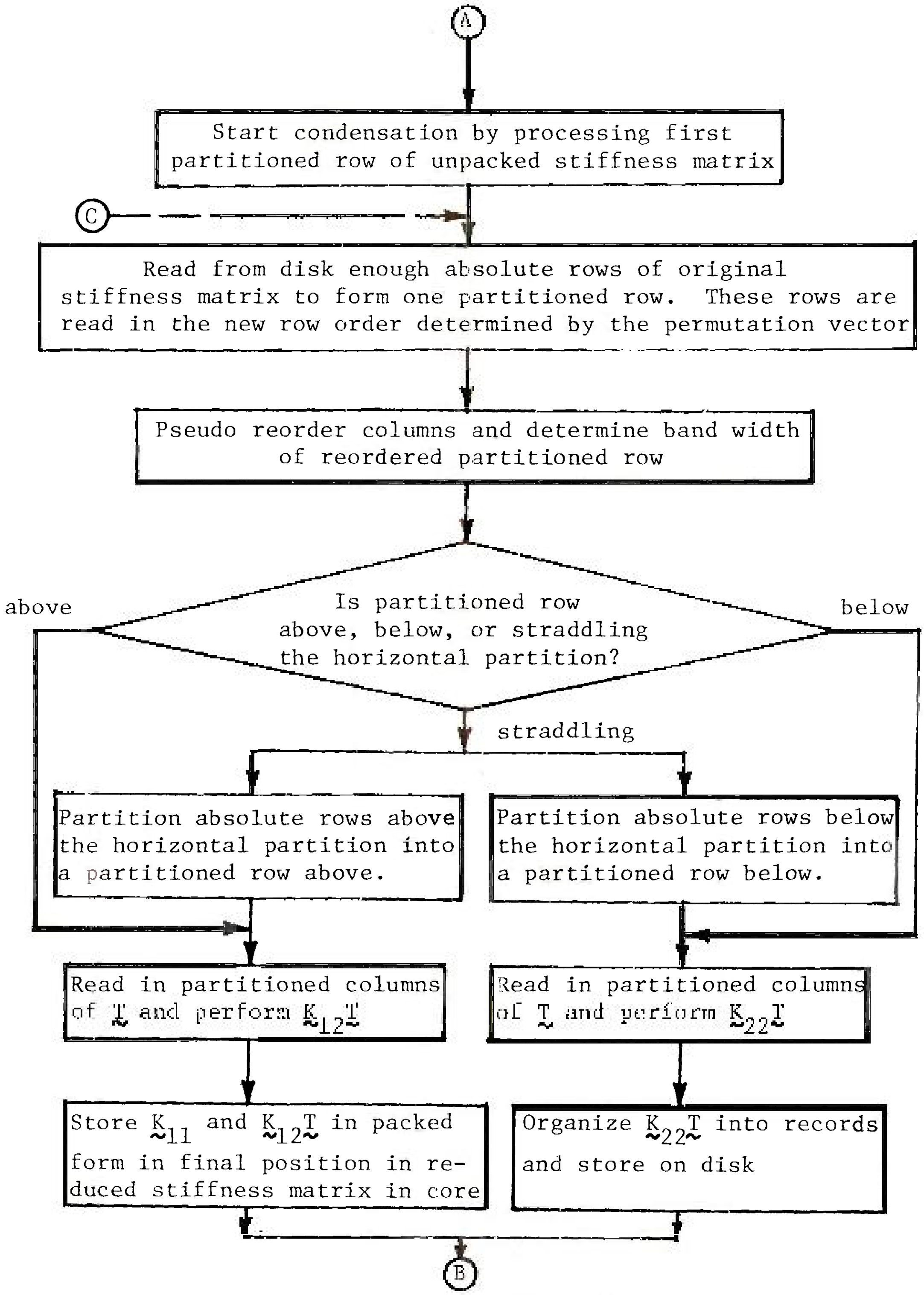


Figure 3.5 Continued

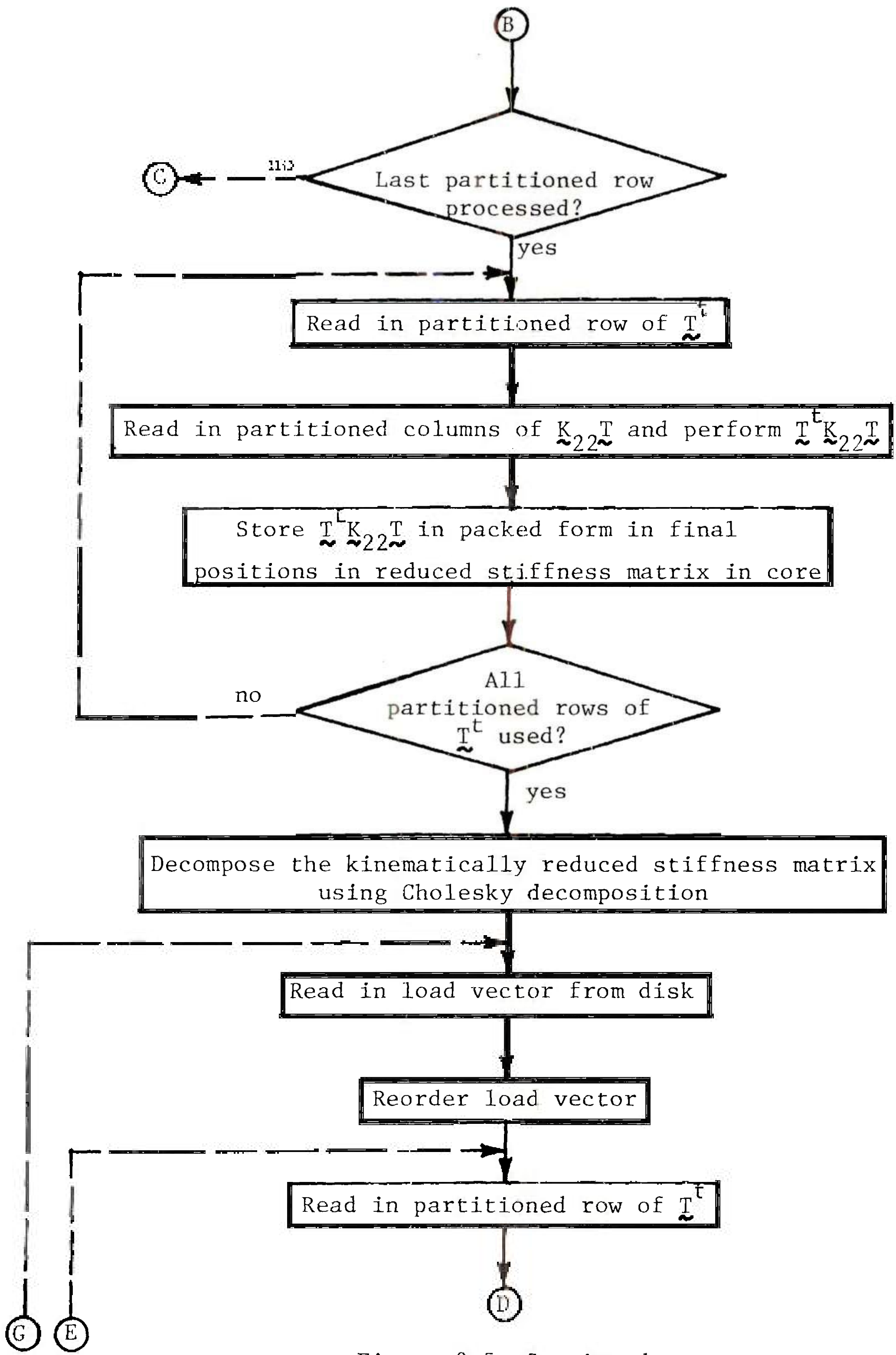


Figure 3.5 Continued

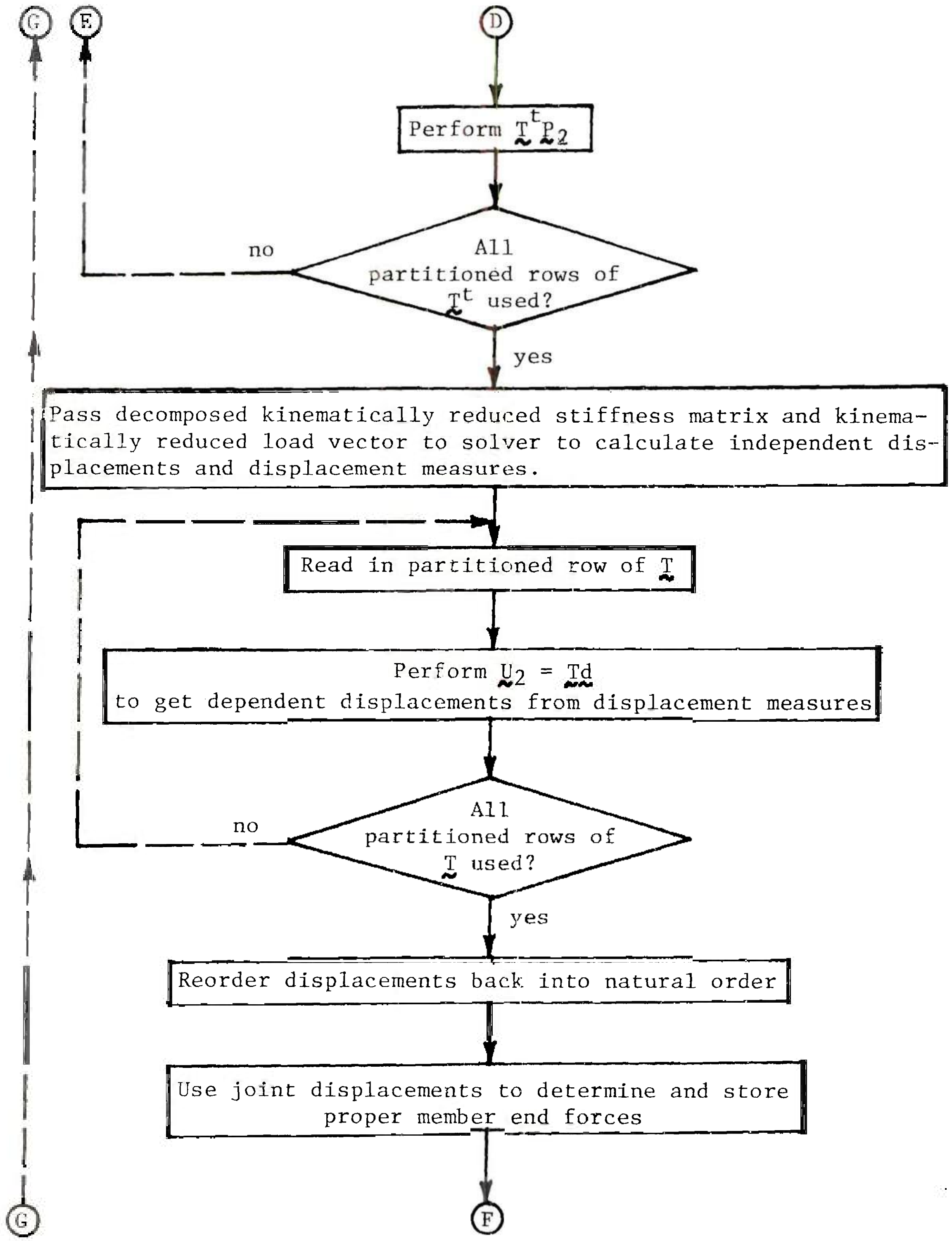


Figure 3.5 Continued

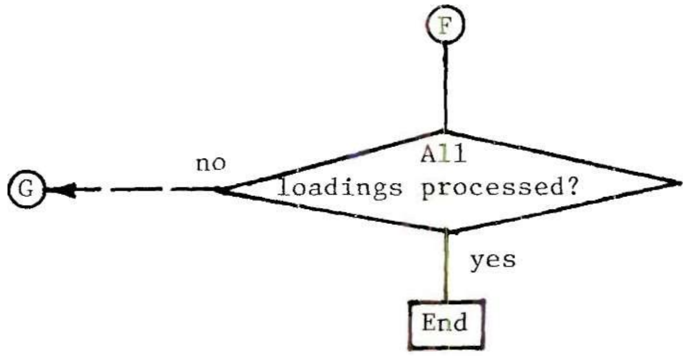


Figure 3.5 Continued

which may be handled. For this reason, the kinematic condensation procedure is set up to be performed on partitioned rows and columns of both $\underset{\sim}{T}$ and $\underset{\sim}{K}$ in an out-of-core procedure. Many references will be made to disk files which are described in Appendix C. The subroutines which execute the kinematic condensation procedure are described in Appendix D. Because this procedure is a part of the total design system, it must be interfaced with a user supplied analysis program. Necessary steps to interface with this program are outlined in Chapter 4.

The kinematic condensation process is broken down to a set of bookkeeping steps, a formation of the stiffness matrix in the proper form to be reduced, and finally the actual condensation of the stiffness matrix and load vectors.

The first step in the procedure, as shown in Fig. 3.5, is to input data and perform consistency checks. Exact format and content of input is detailed in Appendix A. Error messages and program termination result when any data is not within its prescribed bounds. These error messages may be found in the detailed discussion of subroutines.

Among the more important data input is the identification of the dependent displacement components (i.e. their location and direction) in terms of the local displacement directions at joints, as shown in Fig. 3.2(b). All displacement components not so identified are taken to be independent. These displacement components (dependent and independent, but excluding components corresponding to reactions) are all in terms of joint numbers and local direction numbers 1 to 6 (Fig. 3.2). Before processing proceeds, however, these components must be related to the total of n unknown structure displacements. This is done by simply ordering all the dependent and independent components in natural joint order, thereby associating each component with a specific position in the (nx1) displacement vector $\underset{\sim}{U}$ in Eq. 3.1. The $\underset{\sim}{U}$ vector is then reordered into independent components, $\underset{\sim}{U}_1$, and dependent components, $\underset{\sim}{U}_2$, in Eq. 3.2. The reordering, however, is not performed by physically interchanging elements of $\underset{\sim}{U}$. Rather, it is represented by a permutation vector, IOR, which specifies the final position of each displacement component of $\underset{\sim}{U}$ in the reordered $\underset{\sim}{U}_1$ and $\underset{\sim}{U}_2$.

Another important input datum is the $\underset{\sim}{T}$ matrix. This can be input by specifying all of its elements in the proper form as discussed in Section 3.3, or the system may be requested to automatically generate $\underset{\sim}{T}$ based on either an in-plane rigid floor or a completely rigid floor assumption. In any case, once formed, the $\underset{\sim}{T}$ matrix is partitioned and organized into records for storage on a direct access disk file. The partitioning of $\underset{\sim}{T}$ is based upon an arbitrarily selected value of six rows and six columns per row partition and column partition respectively. Note that if the total number of rows and/or columns of $\underset{\sim}{T}$ is not an even multiple of six, then the last partitioned row and/or last partitioned column will contain less than six rows and/or columns respectively.

Records are next created where each partitioned row and column intersect. They are numbered sequentially from left to right across each partitioned row. Due to the selected partitioning, the size of each record is 36 elements, or a 6 by 6 submatrix. However, if the record is in the last partitioned row or column, it may have less than 36 elements.

For example, suppose a $\underset{\sim}{T}$ matrix is developed for the structure of Fig. 3.3, and assuming complete floor rigidity. The form of the $\underset{\sim}{T}$ matrix is that of Eq. 3.16. Each submatrix is of the form of Eq. 3.20. Therefore, the overall $\underset{\sim}{T}$ matrix has 48 rows and 12 columns, and 8 partitioned rows and 2 partitioned columns, as shown in Fig. 3.6. There are also 16 records, each containing 36 elements, and numbered sequentially as shown in Fig. 3.6.

If the $\underset{\sim}{T}$ matrix is generated by the design system, no more than one partitioned row of records is generated in core at any one time. Each new partitioned row is generated in the core space occupied by the previous one, but only after records from the previous row are written to a disk file. If the $\underset{\sim}{T}$ matrix is input by the user, it is input by records in the proper order. As soon as each record is read in, it is immediately written to a disk file. These measures are necessary in order to conserve core storage space in the computer. Note that if the completely rigid assumption is used, records 2, 4, 6, 8, 9, 11, 13 and 15 are all zero, due to the sequential numbering of joints on each floor in the frame as described in Section 3.3.

After storing the complete $\underset{\sim}{T}$ matrix on disk, a single partitioned row of the stiffness matrix is read into core in packed form, an individual row at a time, from the disk file where it was stored by the user supplied analysis program (see Chapter 4), and then it is unpacked into partitioned form. Packed form will be detailed after a discussion of the partitioning of the stiffness matrix, which differs from the partitioning of the $\underset{\sim}{T}$ matrix and is described as follows.

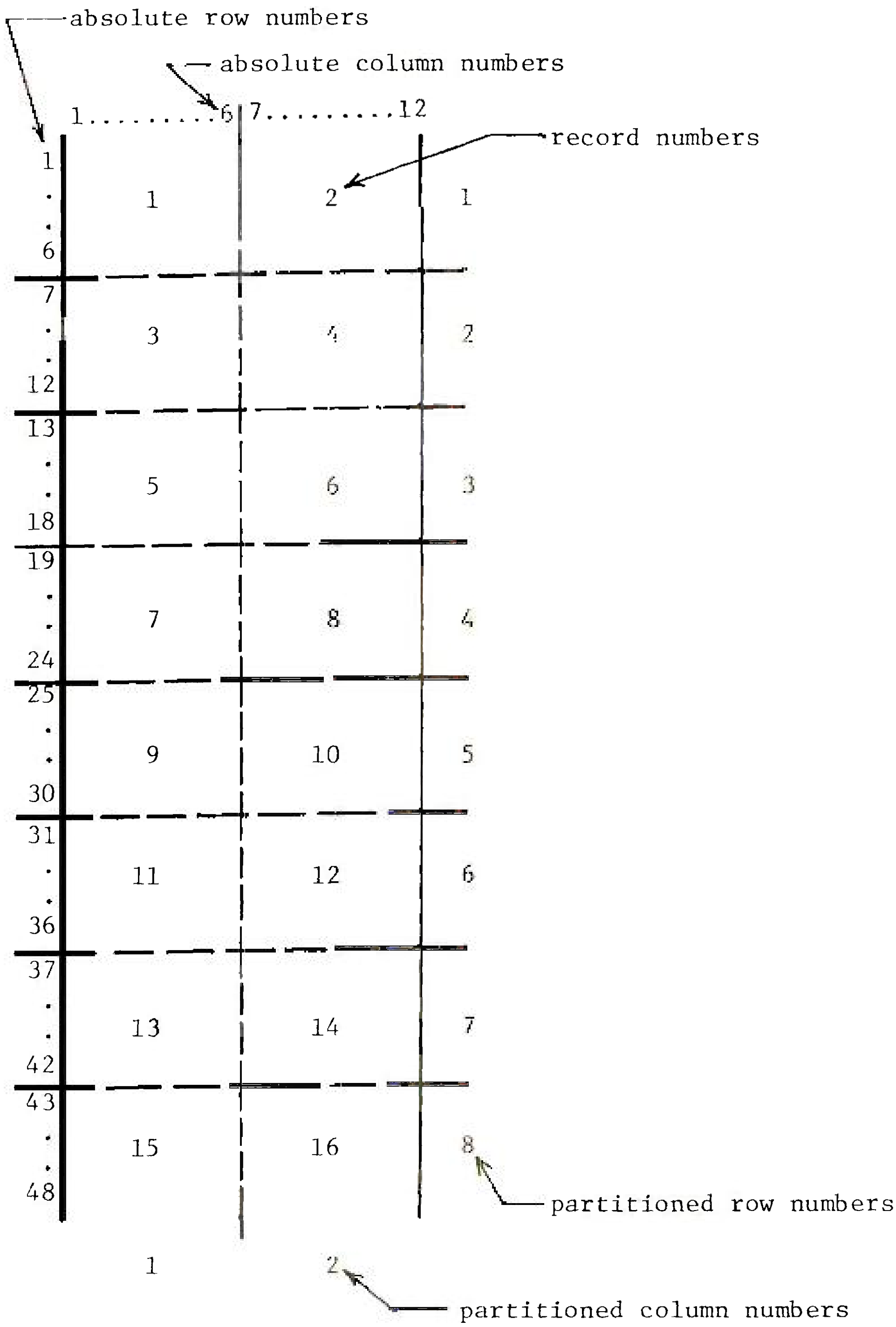


Figure 3.6 The Partitioned $\underset{\sim}{T}$ Matrix, and Partitioned Product $\underset{\sim}{K}_{22}\underset{\sim}{T}$

The stiffness matrix is partitioned into as many rows and columns as there are joints with at least one unknown displacement. Each joint in a space frame has a maximum of 6 unknown displacements. Therefore, since one stiffness equation represents one unknown displacement, a maximum of 6 absolute rows of the stiffness matrix have been chosen to compose one partitioned row. The actual number of rows in a partitioned row is dependent upon the actual number of unknown displacements at each joint. If 4 directions are free to displace (whether they are independent, dependent or any combination thereof), and 2 are restrained, the partitioned row representing this particular joint will have 4 rows. The same applies to partitioned columns of the stiffness matrix. Note that if a joint is fully restrained, there is no partitioned row or column corresponding to that joint. Records of the stiffness matrix are defined by the intersection of partitioned rows and columns, but they are numbered differently than in the $\underset{\sim}{T}$ matrix. To illustrate, suppose that the structure under consideration is the one in Fig. 3.3. Since there are 8 joints numbered consecutively as a, b, c, d, p, q, r and s, which have at least one degree of freedom, then the stiffness matrix has 8 partitioned rows and 8 partitioned columns. Each of these joints has 6 unknown displacements as there are no restraints at any of them. Consequently, every partitioned row contains 6 absolute rows and every partitioned column contains 6 absolute columns. A total of 48 absolute rows and columns make up the stiffness matrix for this structure. Fig. 3.7 shows this partitioned stiffness matrix with all absolute row and column numbers, partitioned row and column numbers, and record numbers identified.

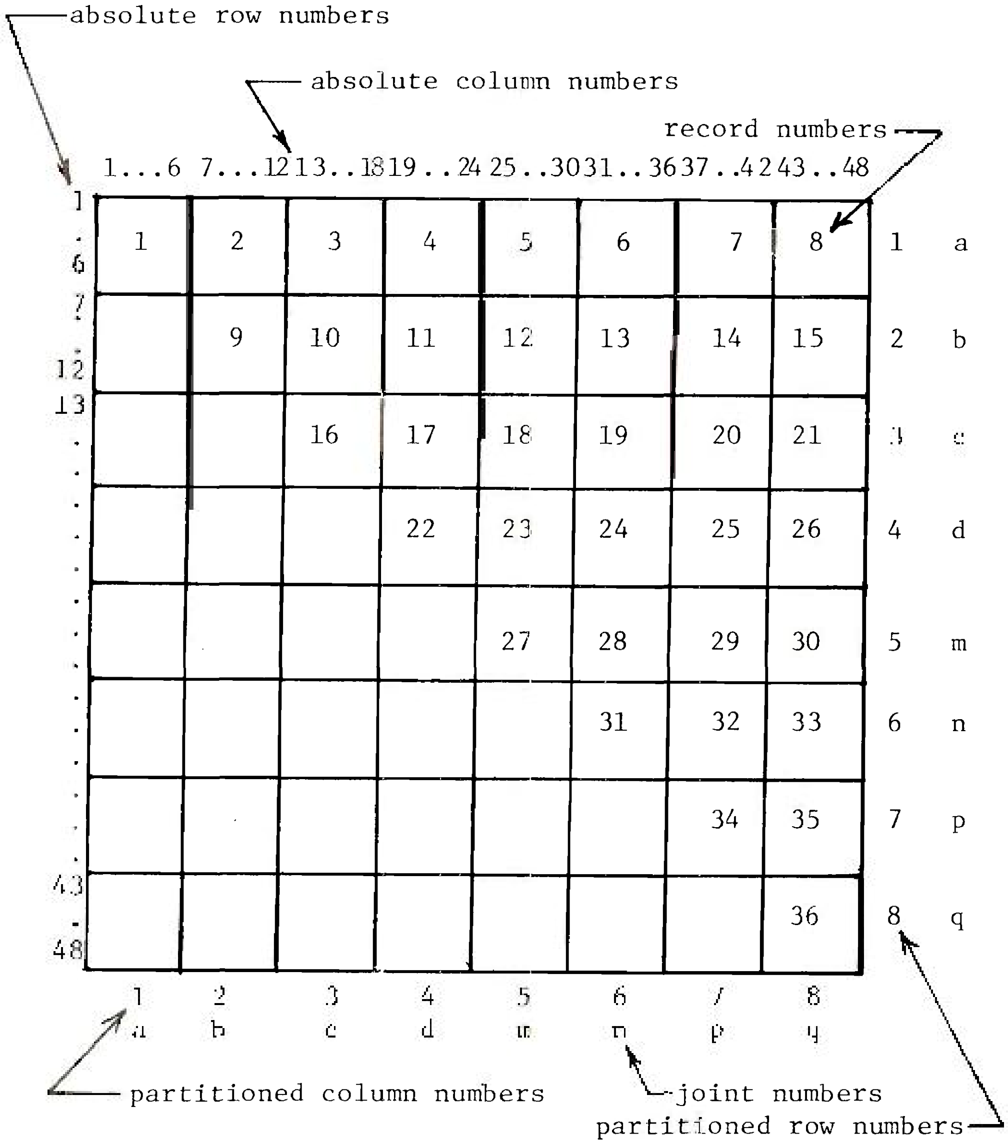


Figure 3.7 Disk Record Organization of Partitioned Original Stiffness Matrix for Structure of Fig. 3.3

Records are numbered across each row, beginning with the diagonal record. The diagonal record in any partitioned row is defined as that record which contains the diagonal elements of each individual row contained in the partitioned row. Each record to the left of the diagonal one may be obtained by simply recognizing the symmetry of the stiffness matrix, and advantage is taken of this fact at a later time. After unpacking,the diagonal records are not written to disk, but are numbered for convenience in accessing records when necessary. Records to the right of the diagonal record, except for ones which contain all zeroes, are written back to disk for subsequent access. This will be described later. The maximum size of any record is 36 elements or a 6 by 6 submatrix. However, the actual number of elements in any record depends upon the number of rows and columns in the partitioned row and column respectively which intersect to form that record. It is important to note that this record structure applies to the unpacked form, which will be described later, as opposed to the packed form, described next.

A discussion of the packed form of the stiffness matrix, as it relates to any single partitioned row which is being processed, is presented here. The packed form stores the elements of the stiffness matrix in an array in such a way where the diagonal elements are placed in column 1, and elements to the right of the diagonal in columns 2 and higher in a sequential fashion in the array. As mentioned earlier, a partitioned row is read into core from disk, one individual row at a time. In order to reduce to a minimum the number of elements which must be transferred, the band width of the original stiffness matrix is employed. The band width is defined as the largest number of columns between the diagonal element and the last nonzero element, inclusive, of any single row in the matrix. It is directly related to the joint numbering and member incidences used to describe the structure. Hence, when rows are read from disk, only those elements between the diagonal and last nonzero element, inclusive, are transferred (note in Chapter 4 that the user supplied analysis program is required to store the rows of the stiffness matrix in this fashion, i.e., packed form taking advantage of the band width).

To illustrate an example of a packed, partitioned row, suppose the second partitioned row of the stiffness matrix relating to the structure in Fig. 3.3 is in core. It is composed of absolute rows 7 to 12, and it has a band width of 24. Fig. 3.8 displays the packed form of partitioned row 2.

Since the stiffness matrix for a linearly elastic structure is symmetric, elements to the left of the diagonal need not be formed. However, for the operation of reordering and reducing, Eqs. 3.2 and 3.8 respectively, elements to the left of the diagonal are needed. Due to symmetry, they may be generated from elements to the right of the diagonal. This, in fact, is the objective accomplished by the next several steps shown in Fig. 3.5, beginning with unpacking the partitioned row of the stiffness matrix under consideration.

Unpacking begins by shifting each row in the packed form to the right until all elements are in their correct column positions relative to the actual stiffness matrix. Note that this process produces a void

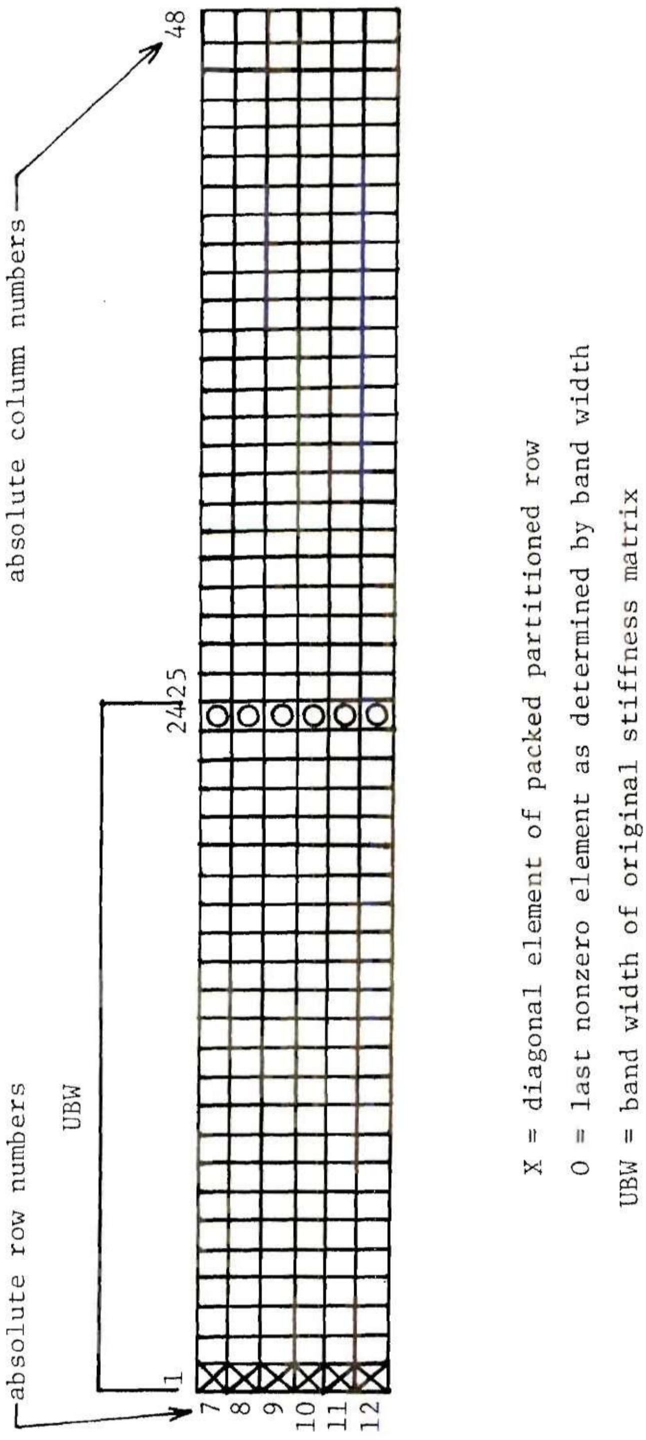


Figure 3.8 Packed Partitioned Row 2 of Original Stiffness Matrix for Structure of Fig. 3.3

to the left of the diagonal which will ultimately be filled.

For example, consider again packed partitioned row 2 of the stiffness matrix relating to the structure of Fig. 3.3, as shown in Fig. 3.8. After unpacking, that row appears as shown in Fig. 3.9. The record numbers correspond to those in Fig. 3.7 for partitioned row 2. The record structure of the stiffness matrix in Fig. 3.7 applies to the unpacked form of the partitioned row as emphasized earlier. Compare and contrast Fig. 3.8, the packed partitioned row and Fig. 3.9, the unpacked partitioned row, observing the void to the left of the diagonal created by the unpacking process. This must be filled.

So, the next step from Fig. 3.5 is to fill in records to the left of the diagonal record in the partitioned row just unpacked for all partitioned rows except the first. For example, consider as before, partitioned row 2 for the structure of Fig. 3.3 in its unpacked form shown in Fig. 3.9. Absolute rows 7 to 12 and absolute columns 1 to 6 compose a record to the left of the diagonal record number 9. Now, refer to Fig. 3.7 and locate these same row and column numbers. Because of the symmetry of the stiffness matrix, this record location may be correctly filled with elements in the transpose of record 2, which is in partitioned row 1. So, record 2, which is stored on a disk file from a previous unpacking operation, is read into core, transposed and placed in the vacant space in partitioned row 2. Observe that record 2 is positioned to the right of the diagonal record in a partitioned row which has already been processed.

At this point, the entire partitioned row is complete except for those elements to the left of the diagonal element in the diagonal

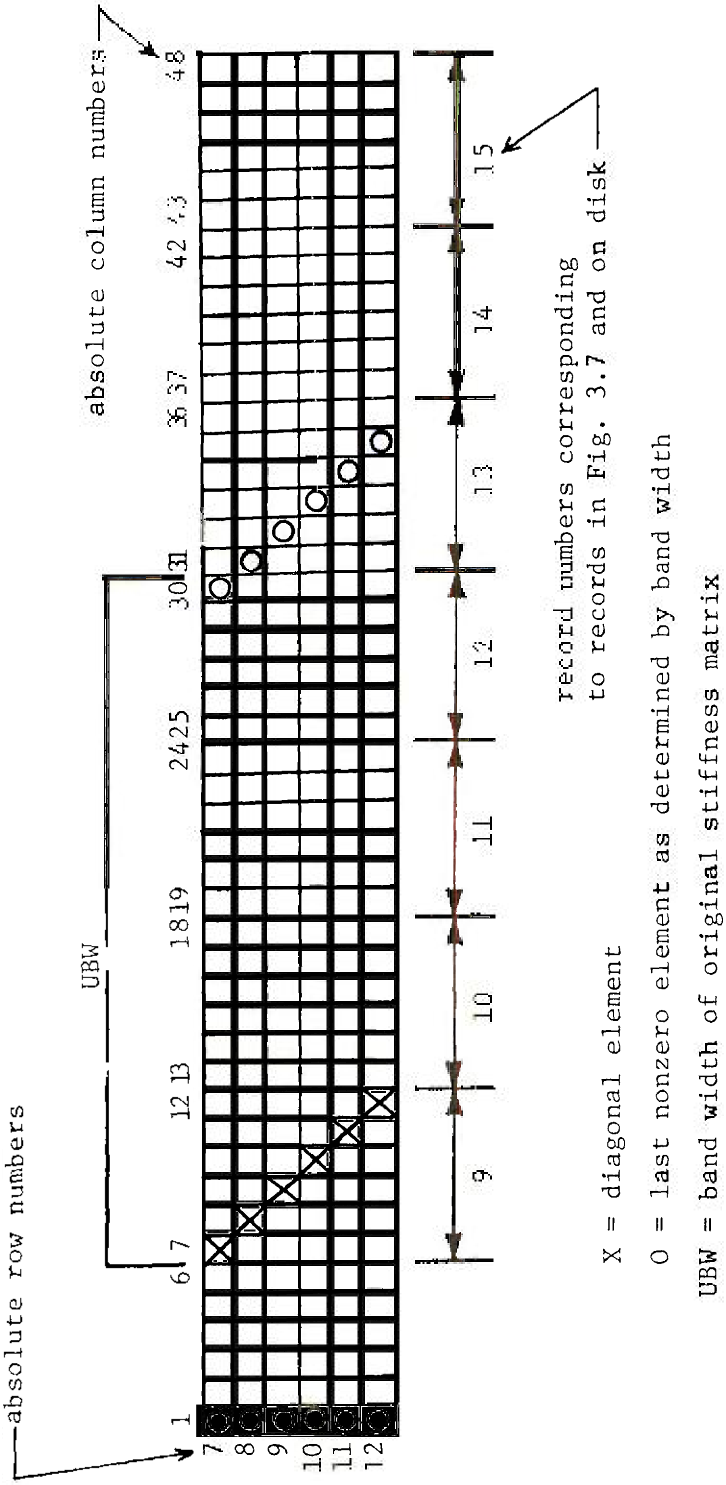


Figure 3.9 Unpacked Partitioned Row 2 of Original Stiffness Matrix for Structure in Fig. 3.3

record, number 9. Due to symmetry, it is simple to generate the missing elements by transposing this diagonal record.

Thus far, all records from the first through the diagonal record in any partitioned row have been accounted for. It is now necessary to write to a disk file all records to the right of the diagonal one, before completing the processing of this partitioned row. These records are used in the processing of subsequent rows to fill in the records to the left of the diagonal. In Fig. 3.9 for partitioned row 2, records 10, 11, 12, 13, 14, and 15 are written out. This procedure is followed for every partitioned row except the last row, since the last row has no records to the right of the diagonal one.

The partitioned row of the stiffness matrix under consideration is complete (unpacked and filled in). Each individual row of the partitioned row is now written to a disk file, since it will be needed subsequently when the kinematic condensation begins. Also note that only one partitioned row is in core at any time during unpacking and filling in to conserve storage space.

The steps of reading in a packed partitioned row, unpacking, filling in to the left of the diagonal, writing the records to the right of the diagonal to disk, and writing the full individual rows to disk, are performed for each partitioned row. These processes produce the stiffness matrix in the form necessary for the actual kinematic condensation.

Fig. 3.5 shows the reducing process begins by reading in from a disk file enough rows of the unpacked stiffness matrix to form a partitioned row. These rows are read in according to the required row reordering as dictated by the permutation vector described earlier.

Immediately following the filling of a partitioned row, which effectively physically reorders the absolute rows, it is necessary to reorder the columns. A pseudo column reordering occurs using the permutation vector to map each element from its old to its new column position. No physical reordering of columns takes place as a result of the mapping procedure. The band width of the reordered partitioned row in question is now determined in order to permit more efficient subsequent matrix operations.

Now, it must be assessed whether the partitioned row is above, below, or straddles the horizontal boundary between independent and dependent equations in the full reordered stiffness matrix. The boundary referred to is shown as the horizontal dotted line below $\underset{\sim}{K}_{11}$ and $\underset{\sim}{K}_{12}$, but above $\underset{\sim}{K}_{21}$ and $\underset{\sim}{K}_{22}$ in Eq. 3.2. Location of the partitioned row relative to this partition is important because it directs the matrix operations to be performed on that row. Of equal importance is the vertical partition in the same equation which performs the function of separating the independent and dependent columns in the full reordered stiffness matrix. These partitions define the 4 submatrices of the reordered matrix in Eq. 3.2, each of which is operated upon in a different fashion as shown in the final equations, Eq. 3.8.

First assume that the partitioned row in question is completely above the horizontal partition which means it lies in $\underset{\sim}{K}_{11}$ and $\underset{\sim}{K}_{12}$. The portion of this partitioned row in $\underset{\sim}{K}_{11}$ remains unchanged, but the portion in $\underset{\sim}{K}_{12}$ must be post multiplied by $\underset{\sim}{T}$. This is accomplished by reading in from disk one partitioned column of $\underset{\sim}{T}$ at a time, beginning with the first, and performing the necessary operations. Each new partitioned column of $\underset{\sim}{T}$ read in will replace the previous one in the identical position in core memory in order to conserve memory space.

To illustrate, refer to the $\underset{\sim}{T}$ matrix of Fig. 3.6. Although this $\underset{\sim}{T}$ was developed for the fully rigid floor case, which means no independent displacements exist and therefore no $\underset{\sim}{K}_{11}$ or $\underset{\sim}{K}_{12}$ exist, it is convenient to use this matrix as an example. The only difference is that it is now assumed that a structure with 48 dependent displacements and 12 displacement measures is being processed. In this case, records 1, 3, 5, 7, 9, 11, 13, and 15 constitute the first partitioned column of $\underset{\sim}{T}$. Once the product of $\underset{\sim}{K}_{12}\underset{\sim}{T}$ is completed for this column, records 2, 4, 6, 8, 10, 12, 14, and 16 are read from disk to complete the full product of $\underset{\sim}{K}_{12}\underset{\sim}{T}$. Note that only elements in $\underset{\sim}{K}_{12}$ which lie within the band width to the right of the diagonal of the partitioned reordered row are used in the matrix operations, since elements outside the band are zero.

Now, for convenience of subsequent discussion, a different structure is assumed; one which has rigid in-plane floors with 24 independent displacements, 24 dependent displacements, and 6 displacement measures corresponding to the structure shown in Fig. 3.3. Fig. 3.10 represents partitioned reordered row 2 for this structure. It should be recognized that although absolute row and column numbers are the same as before, the contents are different due to the reordering. Also, assume the band width for this row is 30. So, row 7, columns 25 to 36

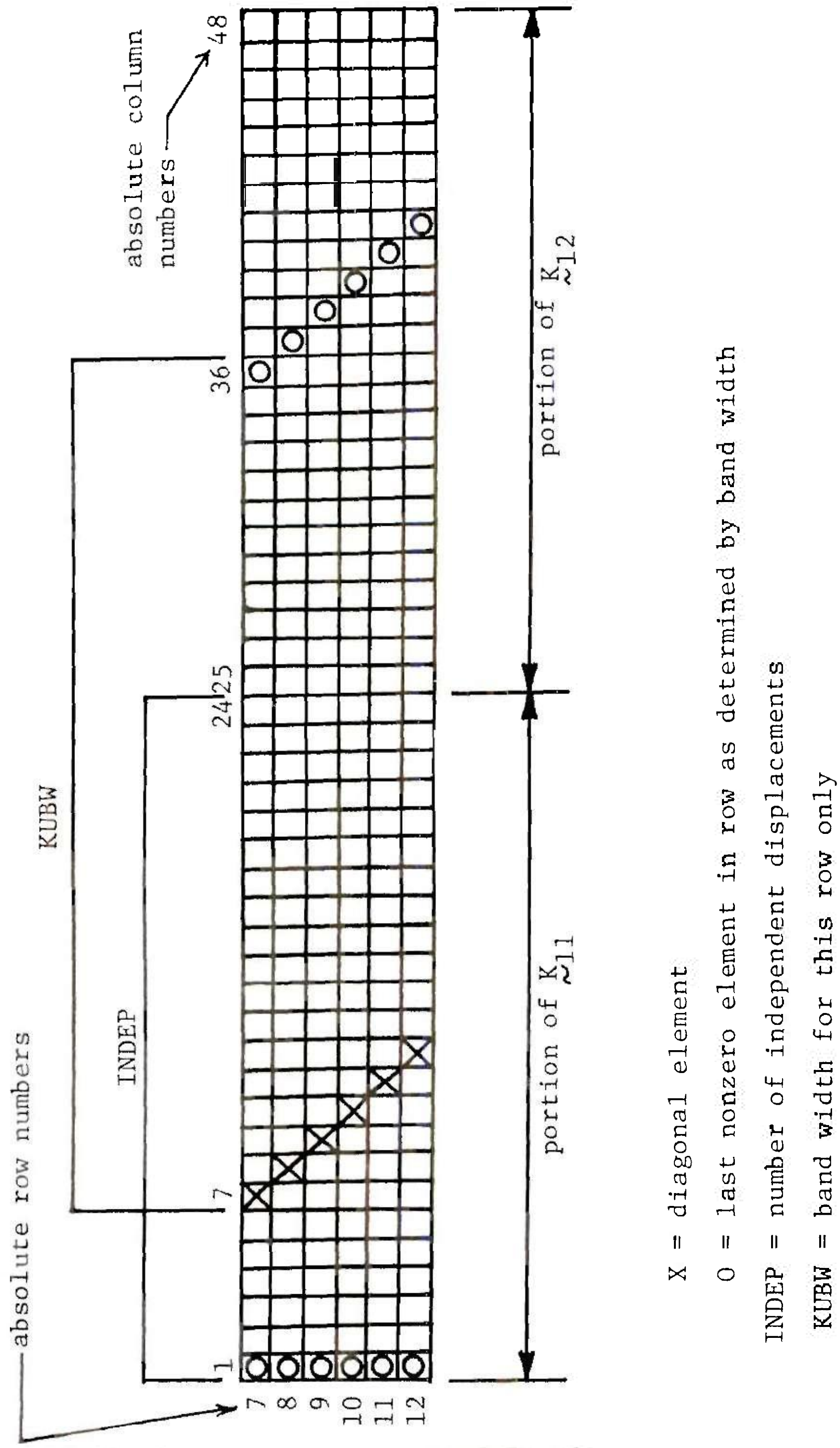


Figure 3.10 Reordered Partitioned Row 2 of Original Stiffness Matrix for Structure in Fig. 3.3

inclusive, and row 8, columns 25 to 37, and so on for each succeeding partitioned row are significant in the matrix operations.

It is important to note that a fully reduced partitioned row now exists in core, following reordering of rows and columns and proper operations by the $\underset{\sim}{T}$ matrix. Any reduced partitioned row under consideration, whether it is above or below the horizontal partition, is now transferred to its final position in a separate array in core which stores the complete final kinematically reduced matrix in packed form (packed form is permitted since the final kinematically reduced matrix is symmetrical -- Eq. 3.8). While the original stiffness matrix is an nxn, the final reduced matrix is (t + r) x (t + r), where t + r is the sum of the independent displacements and displacement measures. At the present time, the maximum value of (t + r) is 360.

Now, assume that the partitioned row in question lies completely below the horizontal partition. This implies it lies in $\underset{\sim}{K}_{21}$ and $\underset{\sim}{K}_{22}$ (Fig. 3.11). Because of the symmetry of the final condensed stiffness matrix, Eq. 3.8, and the fact that this matrix is to be stored in packed form, no operations are performed on the portion of the partitioned row in $\underset{\sim}{K}_{21}$ ($\underset{\sim}{K}_{21} = \underset{\sim}{K}^t_{12}$).

For the portion of the partitioned row in $\underset{\sim}{K}_{22}$, the triple product $\underset{\sim}{T}^t\underset{\sim}{K}_{22}\underset{\sim}{T}$ is performed in two steps which are the post multiplication of $\underset{\sim}{K}_{22}$ by $\underset{\sim}{T}$, followed next by the premultiplication of the product $\underset{\sim}{K}_{22}\underset{\sim}{T}$ by $\underset{\sim}{T}^t$.

The product of $\underset{\sim}{K}_{22}\underset{\sim}{T}$ is performed by reading in one at a time column partitions of $\underset{\sim}{T}$, beginning with the first, exactly as was done

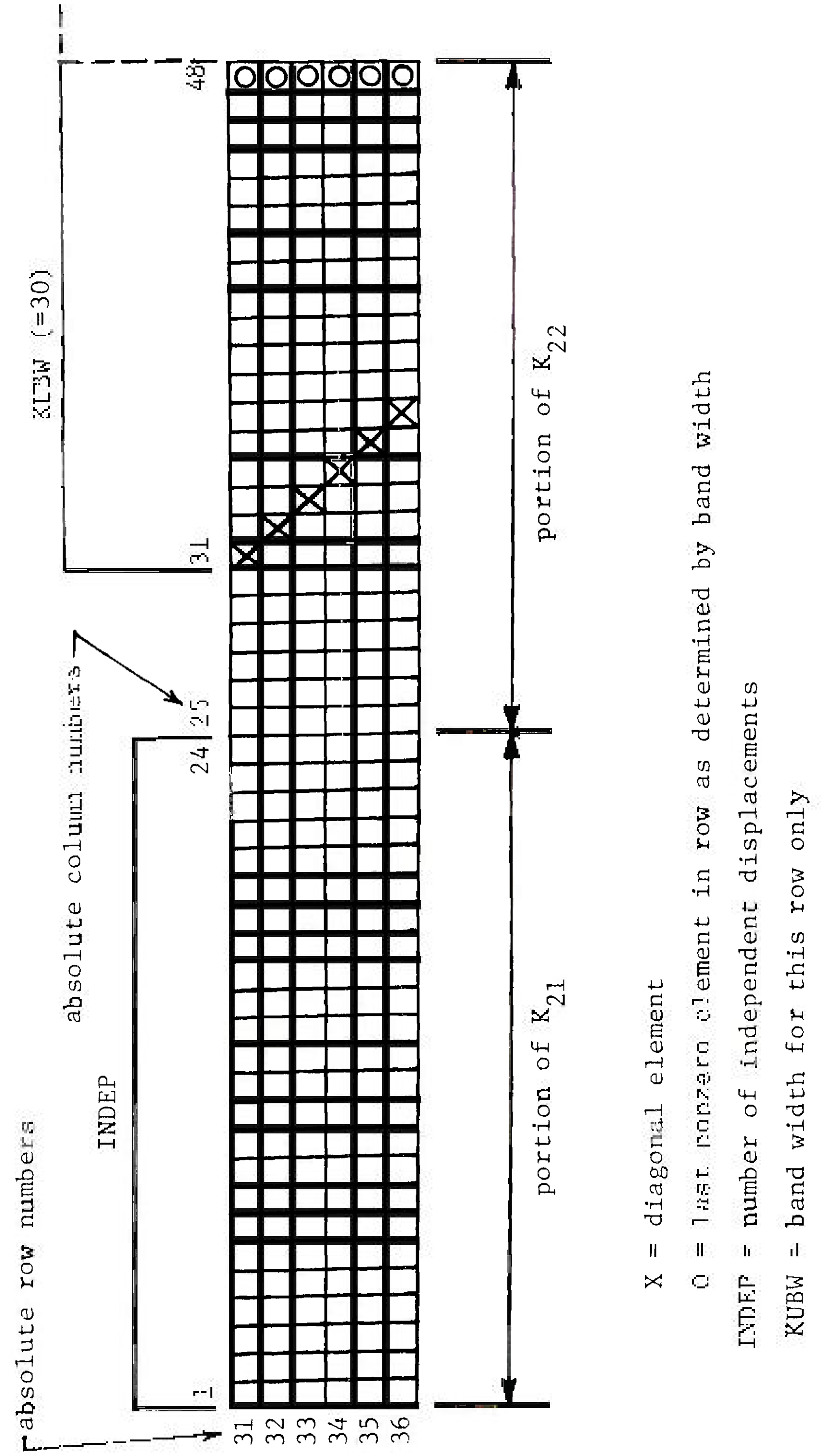


Figure 3.11 Reordered Partitioned Row 6 of Original Stiffness Matrix Completely Below Horizontal Partition of Eq. 3.2 for Structure in Fig. 3.3

for $\underset{\sim}{K}_{12}\underset{\sim}{T}$. Also, as before, the band width of this partitioned row is employed to reduce the number of computations. Those elements which lie within the band width, to the right of the diagonal, plus all elements to the left of the diagonal in $\underset{\sim}{K}_{22}$ of the partitioned reordered row in question, are used in the matrix operations.

The results of $\underset{\sim}{K}_{22}\underset{\sim}{T}$ are organized into records for storage on a disk file in preparation for later premultiplication by $\underset{\sim}{T}^t$. Each partitioned row below the horizontal partition produces a corresponding partitioned row of $\underset{\sim}{K}_{22}\underset{\sim}{T}$ which has the same number of absolute rows as $\underset{\sim}{K}_{22}$ and $\underset{\sim}{T}$ and the same number of absolute columns as $\underset{\sim}{T}$.

At this point, operations on partitioned rows completely above and below the horizontal partition in Eq. 3.8 have been described. A special case occurs when a partitioned row straddles the horizontal partition, which in effect divides the partitioned row into two sub-partitions, one above and one below. Each of these sub-partitions is treated as a full partitioned row where all operations described for rows above and below apply to the sub-partitions of this partitioned row.

The next step is to complete the triple product $\underset{\sim}{T}^t(\underset{\sim}{K}_{22}\underset{\sim}{T})$. Recall that $\underset{\sim}{T}$ and $\underset{\sim}{K}_{22}\underset{\sim}{T}$ are both stored on separate disk files in record form. A single partitioned row of $\underset{\sim}{T}^t$ (i.e. a partitioned column of $\underset{\sim}{T}$ transposed) is read into core beginning with the first. Then successive partitioned columns of $\underset{\sim}{K}_{22}\underset{\sim}{T}$ are read in, one at a time, beginning with the first, each one replacing the previous one in core, to execute the triple product. When all partitioned columns of $\underset{\sim}{K}_{22}\underset{\sim}{T}$ have been operated on, the process starts all over again with the next partitioned row of $\underset{\sim}{T}^t$ which replaces the previous one in core. When all partitioned rows of $\underset{\sim}{T}^t$ have been used, the triple product is complete. Note that only those elements of the triple product $\underset{\sim}{T}^t \underset{\sim}{K}_{22} \underset{\sim}{T}$ which lie on or above the diagonal of the final reduced matrix, Eq. 3.8, are calculated and these are stored in packed form in their final positions, in the full reduced stiffness matrix, as were the results from $\underset{\sim}{K}_{11}$ and $\underset{\sim}{K}_{12} \underset{\sim}{T}$ described previously.

At this point, the complete kinematically reduced stiffness matrix, Eq. 3.8, exists in core memory in packed form. Before proceeding further, this reduced matrix is decomposed by a user supplied Cholesky Decomposition procedure. The decomposed matrix replaces the reduced matrix in core.

It is now necessary to perform a similar kinematic reduction on the load vectors so that they will correspond to the kinematically reduced stiffness matrix. An individual load vector is read into core from a disk file where it was stored by the user supplied analysis program (see Chapter 4). This load vector is reordered by row in accordance with the permutation vector, as was the stiffness matrix.

Eq. 3.8 shows that after reordering, the dependent portion of the load vector, $\underset{\sim}{P}_2$, is to be premultiplied by $\underset{\sim}{T}^t$. So, a single partitioned row of $\underset{\sim}{T}^t$, or column of $\underset{\sim}{T}$ transposed, is read into core and the matrix operation $\underset{\sim}{T}^t \underset{\sim}{P}_2$ is performed in the same way it was done for operations on the stiffness matrix. This continues until all partitioned rows of $\underset{\sim}{T}^t$ are used, resulting in the reduced load vector.

Now that the decomposed kinematically reduced stiffness matrix and kinematically reduced load vector exist, they are passed to the user provided solver (see Chapter 4) to compute independent displacements $\underset{\sim}{U}_1$ and displacement measures, $\underset{\sim}{d}$. Recall that the sum of the independent displacements, $\underset{\sim}{U}_1$, and displacement measures, $\underset{\sim}{d}$, is less than the original total number of joint displacements, n.

The dependent displacements, $\underset{\sim}{U}_2$, may now be computed using the computed values of displacement measures, $\underset{\sim}{d}$, by Eq. 3.3 which is, $\underset{\sim}{U}_2 = \underset{\sim}{T}\underset{\sim}{d}$. This operation is performed by multiplying a partitioned row of $\underset{\sim}{T}$, read from disk, by $\underset{\sim}{d}$ to generate the corresponding elements in $\underset{\sim}{U}_2$. This process is repeated for all partitioned rows of $\underset{\sim}{T}$, after which all dependent displacements in $\underset{\sim}{U}_2$ are known.

The full displacement vector $\underset{\sim}{U}$, corresponding to a particular loading, is now known, but is in the partitioned and reordered form shown in Eq. 3.2, with the independent displacements $\underset{\sim}{U}_1$ followed by the dependent displacements $\underset{\sim}{U}_2$. However, in order to calculate the member forces for use in virtual work computations later, it is necessary to reorder the displacement vector $\underset{\sim}{U}$ back into its original order. This is done by using the permutation vector.

Using the displacements in their natural order, the user supplied analysis program calculates the member end forces, which are necessary for virtual work computations (see Chapter 2).

Each cycle, beginning at the point where a load vector is read in from disk, and ending at the point where member end forces are computed, is repeated for each load vector (both P and Q loads) input by the user. Once all loads are processed and all member end forces for all P and Q loads are computed, the kinematic condensation procedure is concluded. It is important to note that this procedure may be executed many times during the course of a design, except that load vectors are kinematically reduced only once and stored for reuse on subsequent passes. The exact number of times is determined by the user as described in Chapter 2.

CHAPTER 4

INTERACTION WITH USER SUPPLIED ANALYSIS PROGRAM

4.1 Introduction

The total computer stiffness design system consists of essentially three parts as shown in Fig. 4.1, which are the MAIN program, the stiffness design subprograms, and the user supplied stiffness analysis subprograms.

The MAIN program was written to control the overall iterative process of analysis and design.

The stiffness design subprograms were written to control all phases of design (i.e., member size changes), as well as perform the virtual work analyses, and kinematic condensation analyses. As can be seen in Fig. 4.2, kinematic condensation is actually called by the modified user supplied analysis control program (MODFR2).

The user supplied analysis subprograms are a package of programs which a user wishes to use to perform the exact stiffness analyses of the frame.

The user supplied analysis subprograms must, however, be modified so that they can properly interact with the MAIN program and the stiffness design subprograms. To illustrate the required modifications, this stiffness design system implements a stiffness analysis program written by Weaver (15), and called FR2. The FR2 program was modified and assigned a new name, MODFR2.

The modifications contained in MODFR2 and other interactions are described in this chapter. Fig. 4.2 shows the calling sequences between

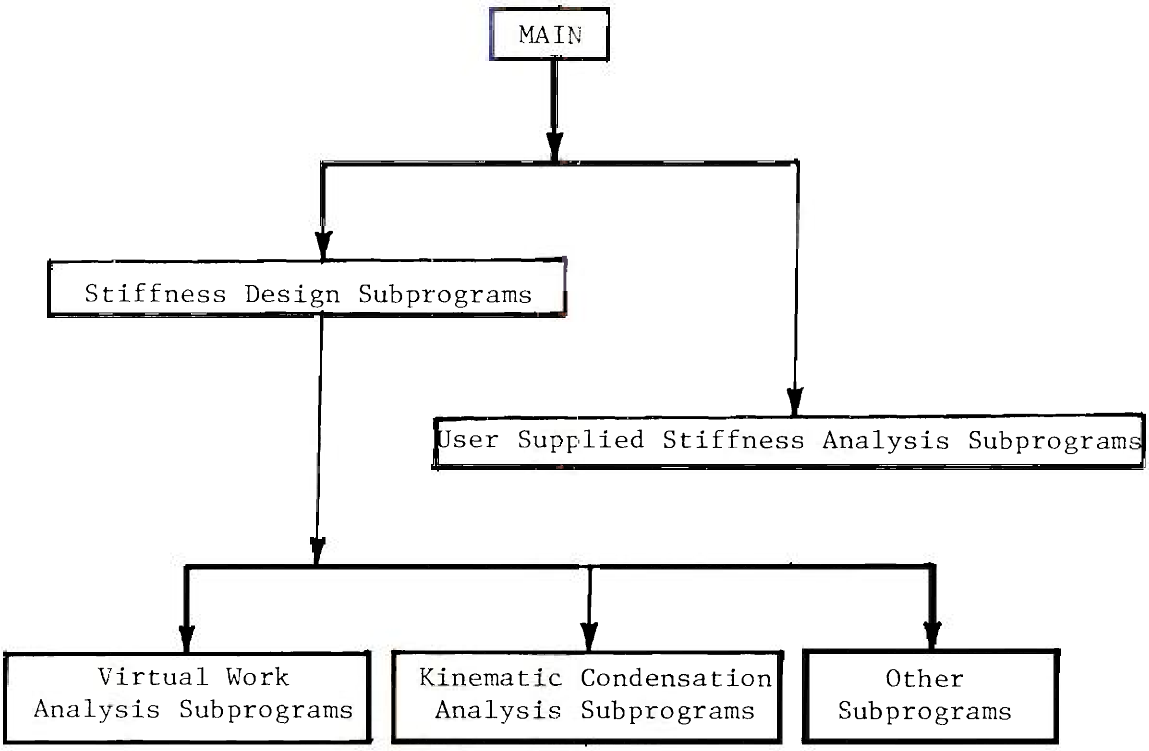
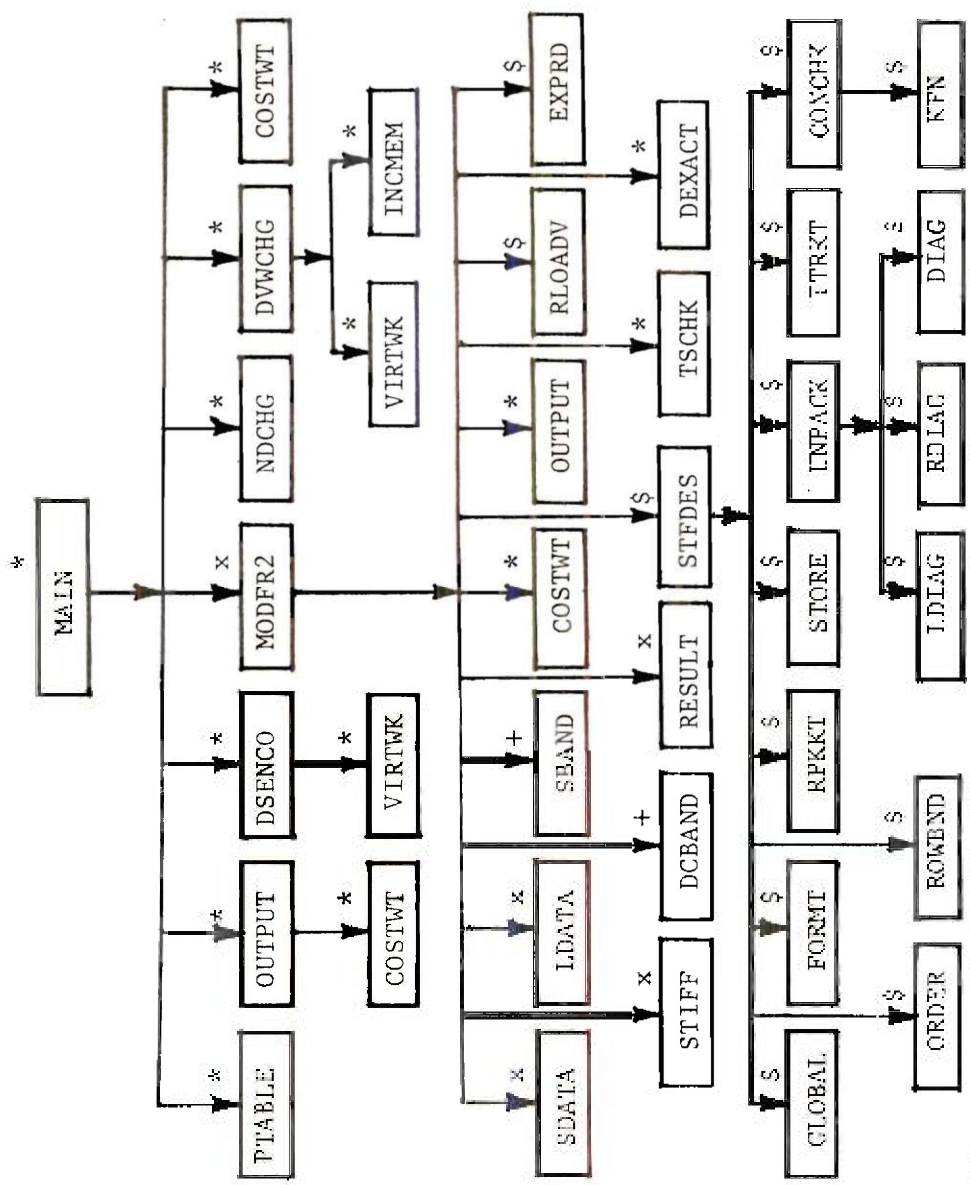


Figure 4.1 Functional Flow Chart of the Computer Stiffness Design System



* = Design subprograms written entirely for the design system.
$ = Kinematic condensation subprograms written entirely for the design system.
x = User provided stiffness analysis subprograms, modified for the design system.
\+ = User provided stiffness analysis subprograms, not modified for the design system.

Figure 4.2 Macro Flow Chart of Subprogram Control all subprograms in the total system.

All subprograms shown in Fig. 4.2 and identified in this chapter are described in Appendix D. In addition, several disk files storing temporary information during analysis and design are referenced in this chapter, and fully described in Appendix C. Furthermore, references are also made to sequential statement numbers in various subprograms. These may be found in the subprogram listings in Appendix F.

4.2 Design System and User Supplied Program Interaction

Subprogram MODFR2 is the modified form of the stiffness analysis control program supplied by the user. It was modified so that in addition to controlling stiffness analysis, it also controls kinematic condensation analysis. Six other subprograms were also provided by the user for stiffness analysis. Two of these, DCBAND and SBAND, were not modified, while the other four, SDATA, STIFF, LDATA, and RESULT, were modified. Details of the modifications are described in Section 4.3.

An overview of the analysis process controlled by MODFR2 is presented next in this Section (also refer to Fig. 4.2). It should be noted that the functional aspects of the process are important, rather than the specific way the subprograms perform the functions. Consequently, any set of user supplied analysis subprograms may be used as long as they are modified in a similar way as the user supplied subprograms described herein are, functionally the same, and operate on the functions in the order required.

Certain types of data required for analysis are input by subprogram SDATA. This includes joint coordinates, joint support data, member incidences and orientations, material properties, and cost and weight factors. Member properties are obtained from disk files containing section tables which were generated by subprogram PTABLE. SDATA also outputs the member length, density, and cost factor to disk file 19 for all members, and joint coordinates to disk file 20 for all joints.

Other types of data required for analysis include loading information. Subprogram LDATA reads this data in and outputs the resulting loading vectors, corresponding to loading conditions, to disk file 15, one record per loading condition.

Subprogram TSCHK, not provided by the user, is called to perform consistency checking on the existence of appropriate sections and section tables. Subprograms OUTPUT and COSTWT, also not provided by the user, are called to output all member properties as they were taken from the section tables, and cost and weight data respectively.

The structural stiffness matrix is assembled by user provided subprogram STIFF. If a kinematic condensation is not to be executed, the stiffness matrix is retained in core memory and not written to disk. Otherwise, the stiffness matrix is written to disk file 11, one row per record. It should be noted that since the stiffness matrix is symmetrical and banded, only those elements from the diagonal element to the last element within the band of the matrix is actually written to disk.

If stiffness analysis is executed, the stiffness matrix is reduced for solution purposes by user provided subprogram DCBAND. The load vectors are reduced, and backsubstitution performed by user provided subprogram SBAND in order to compute all joint displacements. Member end forces and support reactions are then computed by user provided subprogram RESULT. In addition, RESULT writes member axial force and end bending moments to disk file numbers (20 + LN), where LN is the loading condition number.

Displacements corresponding to displacement constraints which are not parallel to the global coordinate system, and floor rotations corresponding to floor rotation constraints are computed by subprogram DEXACT, not provided by the user.

If kinematic condensation analysis is executed, the structural stiffness matrix is kinematically reduced by subprogram STFDES, and all other subprograms referenced by STFDES, none of which are provided by the user. Subprogram DCBAND is then used as before to reduce the kinematically reduced stiffness matrix for solution purposes.

Subprogram RLOADV, not provided by the user, then kinematically reduces the load vectors and writes these results to disk file 15, following the original load vectors, in record numbers (NLS + LN), where NLS is the total number of loading conditions, and LN is a loading condition number.

User provided subprogram SBAND is also used to reduce the kinematically reduced load vectors for solution purposes, and backsubstitute to compute the displacements associated with the kinematically reduced stiffness equations. These displacements, however, are only the independent displacements and displacement measures in reordered form. Subprogram EXPRD, not provided by the user, computes the dependent joint displacements from the displacement measures, and then reorders the independent and dependent joint displacements into their original natural order. Subprograms RESULT and DEXACT are then executed as described before.

4.3 Detailed Modifications to User Supplied Analysis Programs

Detailed coding changes to the user provided programs may be found in the program listings in Appendix F. Locations and general purpose of these changes are outlined in Tables 4.1, 4.2, 4.3, 4.4 and 4.5 for user provided subprograms MODFR2, SDATA, STIFF, LDATA, and RESULT, respectively.

In addtion to coding changes, certain changes to the common data structure are required. The simplest way of determining the required changes is to review COMMON of the example user provided analysis control subprogram MODFR2, COMMON of the kinematic condensation control subprogram STFDES, and COMMON of the design system MAIN control program. In addition, the definition of all COMMON variables which are used by the design system subprograms, not provided by the user, as well as all COMMON variables used by the user supplied subprograms, but only those which are also used by the design system subprograms, are found in Appendix E and should be closely reviewed.

Table 4.1 Modifications Applied to User Supplied Subprograms FR2 to Generate MODFR2

(Note: Line numbers refer to program listing in Appendix F)

Modification Number	Line Number(s)	Purpose of Modification
1	0007	Skip modifications 2 thru 6 on second and subsequent passes (IFIRST = 1).
2	0008	Call TSCHK to insure initial table and section numbers for each member are within proper bounds.
3	0009	Should an error exist in TSCHK, (IERR = 1), skip to end of subprogram and return to MAIN program.
4	0010	Call OUTPUT to print initial member properties.
5	0011-0016	Print heading for initial weight and cost output.
6	0017	Call COSTWT to calculate and print initial weight and cost data.
7	0020	Skip LDATA on second and subsequent passes (IFIRST = 1). (Note: LDATA must be executed once for each loading condition during the first analysis.)
8	0027	Skip modifications 9 thru 19 if a regular analysis is required (IFLAG = 0).
9	0028	CALL STFDES, to construct packed, banded, reduced stiffness matrix.
10	0029	If an error is found in STFDES (IERR = 1), skip to end of subprogram and return to MAIN program.
11	0030	CALL DCBAND to decompose by Cholesky method, the packed, banded, reduced stiffness matrix. (Note: The Cholesky method

Table 4.1 Continued

Modification Number	Line Number(s)	Purpose of Modification
		requires two subroutines to complete, one of which is DCBAND. Should a one step solver be used, it would be called at a later time)
12	0033	Skip modification 13 if one the second or subsequent pass through STFDES, (ISTFD > 0)
13	0034	CALL RLOADV to reduce load vector.
14	0036	READ in a reduced load vector.
15	0037	CALL SBAND to solve for independent displacements and displacement measures, using decomposed matrix from modification 11, and reduced load vector from modification 13. (Note: A one step solver may be inserted here, using packed, banded, reduced matrix with reduced load vector.)
16	0038	CALL EXPRD to expand reduced displacement vector to actual joint displacements, and reorder back into original sequence.
17	0041	CALL RESULT to compute and store member end actions.
18	0042	CALL DEXACT to compute exact value of translational constraint not parallel to global X, Y, or Z axes; and to compute floor rotation if such a constraint exists. (Note: Steps 12 thru 18 are to be carried out for each load vector unless a one step solver is used)
19	0044	Skip to end of subprogram after processing all load vectors.
20	0045	CALL DCBAND for Cholesky decomposition on

Table 4.1 Continued

Modification Number	Line Number(s)	Purpose of Modification
		original stiffness matrix. (Note: The Cholesky method requires two subprograms to complete, one of which is DCBAND. Should a one step solver be used, it would be called later.)
21	0048	READ in unmodified load vector.
22	0049	CALL SBAND to solve for actual joint displacements. (Note: One step solver may be inserted here, using original stiffness matrix and load vectors.
23	0054	CALL RESULT to compute and store member end actions.
24	0055	CALL DEXACT to compute exact value of translated constraint not parallel to global X, Y, or Z axes and to compute floor rotation if such a constraint exists. (Note: Steps 21 thru 24 are to be performed for each load vector unless a one step solver is used.)
25	0060	Skip to end of subprogram.

Table 4.2 Modifications Applied to User Supplied Subprogram SDATA

(Note: Line numbers refer to program listing in Appendix F.)

Modification Number	Line Number(s)	Purpose of Modifcation
1	0006	Skip modification 2 on second and subsequent passes.
2	0082	Write joint coordinates to File 20 immediately after input, record number being equal to the joint they represent.
3	0150	Skip modification 4 on the second and subsequent passes.
4	0153-0154	Input table, section, density factor, and cost factor for each member.
5	0155-0157	READ section properties into core for each member from disk files, based on table and section numbers.
6	0169	Skip modification 7 on second and subsequent passes.
7	0176	Write density factor, cost factor and length, in that order, to File 19.
8	0176	Skip output of member incidences, table, section, etc. on second and subsequent passes.
9	0233	Skip output of joint restraint information on second and subsequent passes.

Table 4.3 Modifications Applied to User Supplied Subprogram STIFF

(Note: Line numbers refer to program listing in Appendix F.)

Modification Number	Line Number(s)	Purpose of Modification
1	0537	Skip modification 2 if a stiffness analysis is requested (IFLAG = 0).
2	0538-0540	Write out each individual row of the original stiffness matrix to File 11. To minimize data transfer, only elements from the diagonal to the last element in a row within the band are written to disk.

Table 4.4 Modifications Applied to User Supplied Subprogram LDATA

(Note: Line number refers to program listing in Appendix F.)

Modification Number	Line Number	Purpose of Modifcation
1	0141	Write out each individual load vector to File 15, where record numbers correspond to loading number LN.

Table 4.5 Modifications Applied to User Supplied Subprogram RESULT

(Note: Line numbers refer to program listing in Appendix F.)

Modification Number	Line Number(s)	Purpose of Modification
1	0279,0280	Write out in order the axial force, y- and z-axis bending moments at negative inciden end, and y- and z-axis bending moments at positive incidence end of members to Files 20 + LN. Record numbers are equal to the member number for which these forces apply.

CHAPTER 5

SUMMARY OF RESULTS

5.1 Introduction

This chapter contains example problems illustrating results of executing the stiffness design system. Two separate frame geometries, designated as Frames A and B, have been selected, one a plane frame and one a space frame. Although the system is only designed for space frames, the plane frame can be modeled and run as a space frame with no out of plane loads or deflections. However, this is not very efficient because all out of plane deflections are calculated, even though it is known they are zero. Solution time is about the same for a space frame with the same number of degrees of freedom. It is expected that a future version of the design system will be able to design plane frames as plane frames.

Design data, geometry, load data, assumptions and results are all presented in succeeding sections of this chapter.

5.2 Frame A, Plane Frame - General Design Data

General design data applicable to Frame A, Fig. 5.1, are presented in this section. Additional data may be found in Section 5.4.

1. Geometrical Conditions

   Figure 5.1 shows the geometry, member and joint numbers and joints of constraint for Frame A.

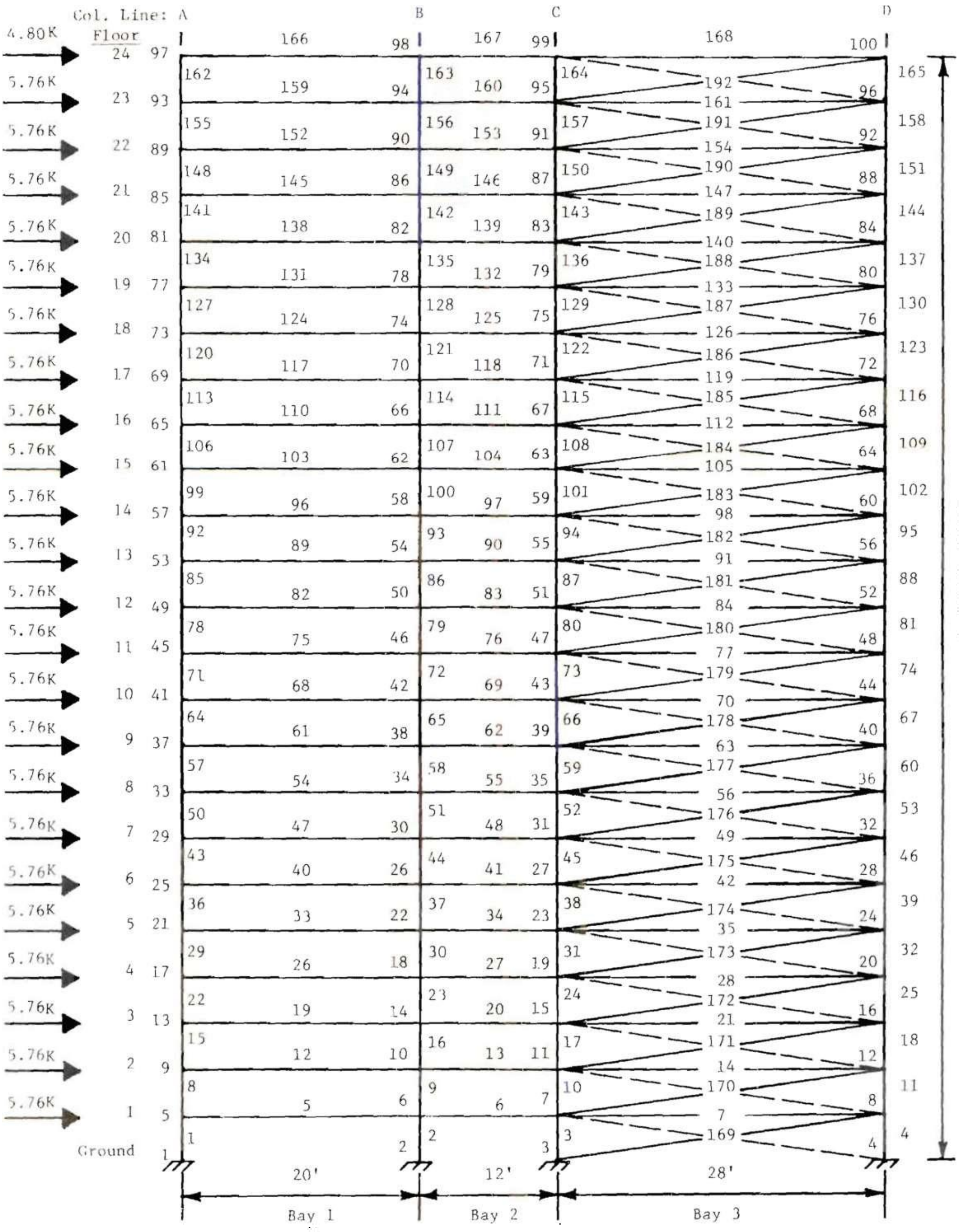


Figure 5.1 Frame A

2. Material - Steel

   a. Modulus of Elasticity, $E = 2.9 \times 10^4$ kips/in.^{2}

   b. Shear Modulus, $G = 1.16 \times 10^4$ kips/in.^{2}

   c. Unit Cost Factor, each member, \$0.20/lb. of steel

   d. Density Factor, each member, 490 lbs./ft.^{3}

3. Load Conditions - Wind only applied as concentrated lateral joint loads as shown in Fig. 5.1.

4. Displacement Constraint Locations and Directions

   a. For Group 1 Examples A1, A2, and A3 (Wind from left and right) Joints 69, 97, and 100, horizontal in plane of the frame.

   b. For Group 2 Examples AWL1, AWL2, and AWL3 (Wind from left) Joints 69 and 97, horizontal in plane of the frame.

   c. For Group 3 Examples AWR1, AWR2, and AWR3 (Wind from right) Joints 72 and 100, horizontal in plane of the frame.

Results for all Frame A examples are shown in Table 5.11.

5.3 Frame B, Space Frame - General Design Data

General design data for Frame B, Figs. 5.2 to 5.8, are presented in this section. Additional data may be found in Section 5.4.

1. Geometrical Conditions

   Figs. 5.2 to 5.8 show the geometry, member and joint numbers, and points of constraint of Frame B.

2. Material - Steel

   a. Modulus of Elasticity, $E = 2.9 \times 10^4$ kips/in.^{2}

   b. Shear Modulus, $G = 1.16 \times 10^4$ kips/in.^{2}

   c. Unit Cost Factor, each member, $0.20/lb. of steel

   d. Density Factor, each member, 490 lbs./ft.^{3}

3. Load Conditions - Wind only applied as concentrated lateral joint loads as shown in Figs. 5.3 to 5.8 for the braced and unbraced cases.

4. Displacement Constraint Locations and Directions

   a. For Groups 4 and 5, Examples B1, B2, B3,BW1, BW2, and BW3: Joint 92, X-direction, Joint 90, Z-direction.

   b. For Group 6 Examples BRW1, BRW2 and BRW3: Joint 94, X-direction, Joint 96, Z-direction.

Results for all Frame B examples are shown in Table 5.12.

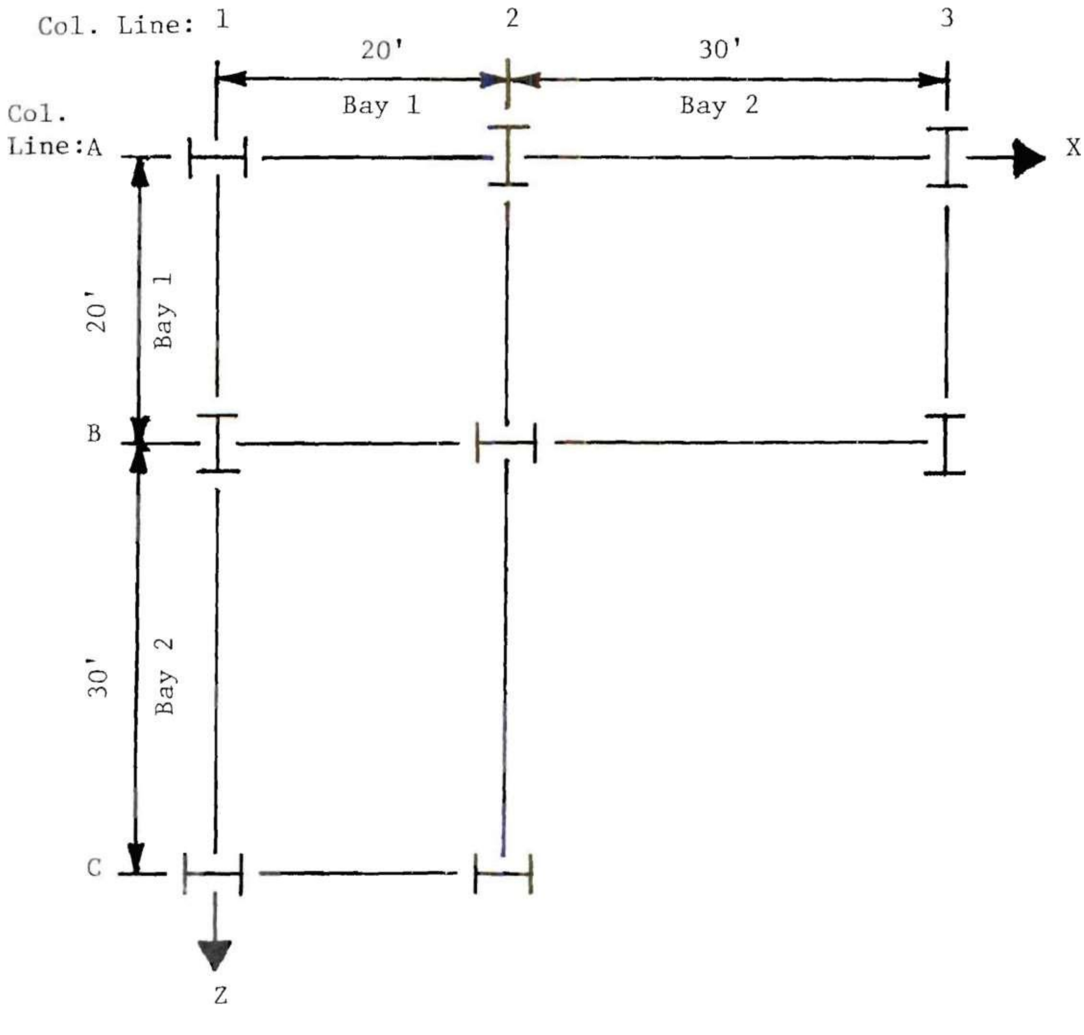


Figure 5.2 Floor Plan for Frame B

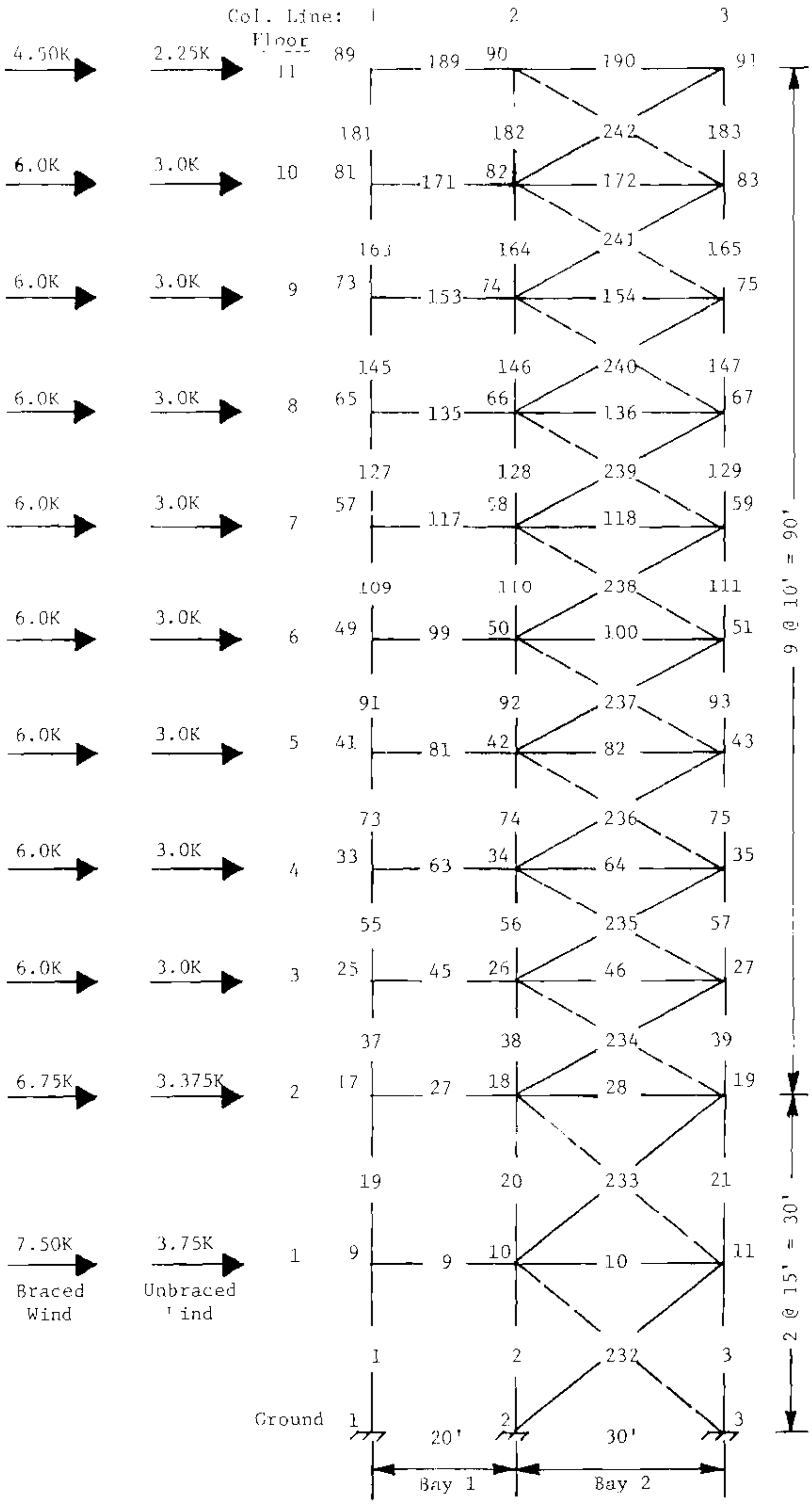


Figure 5.3 Frame B, Column Line A

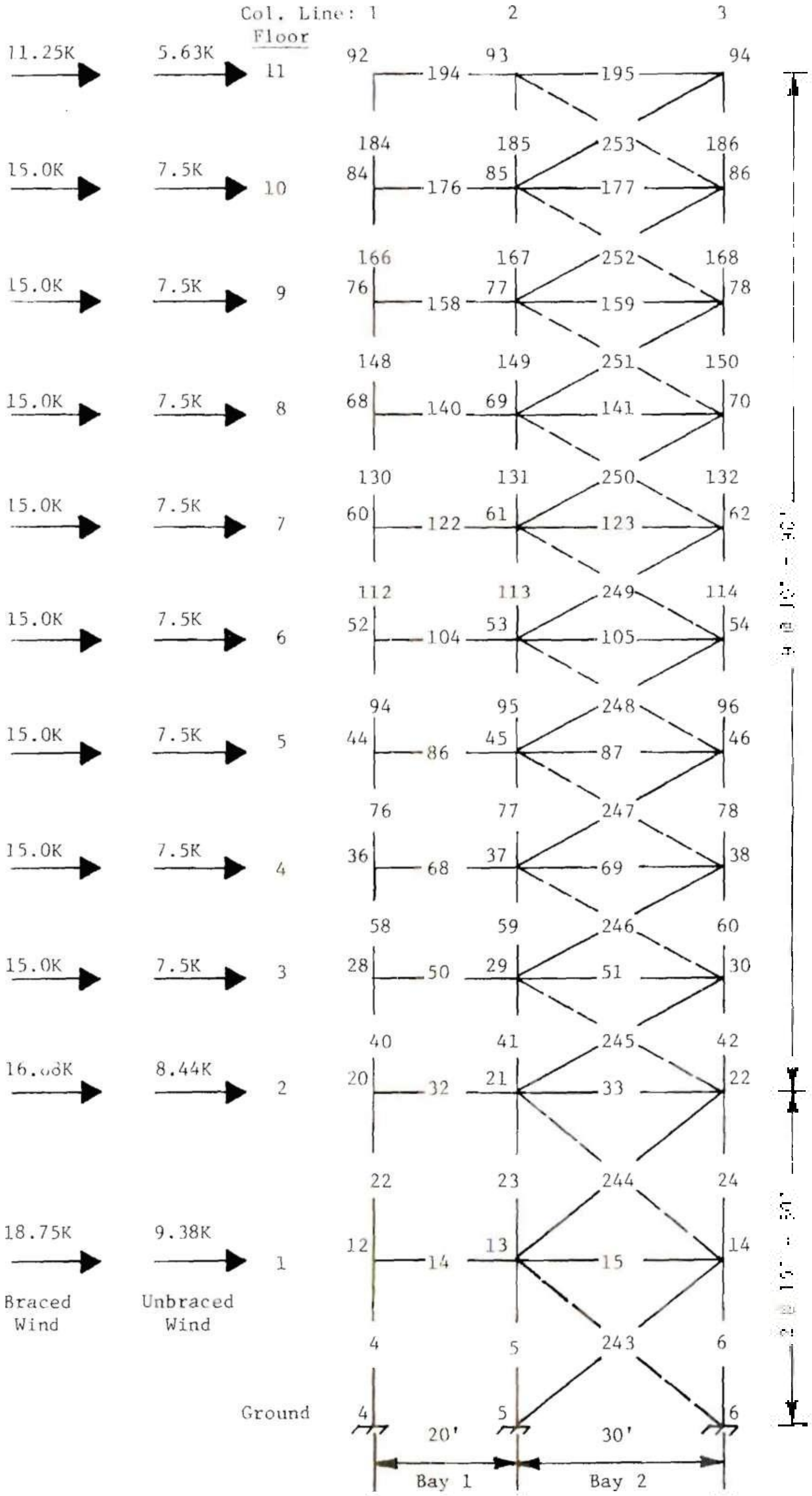


Figure 5.4 Frame B, Column Line B

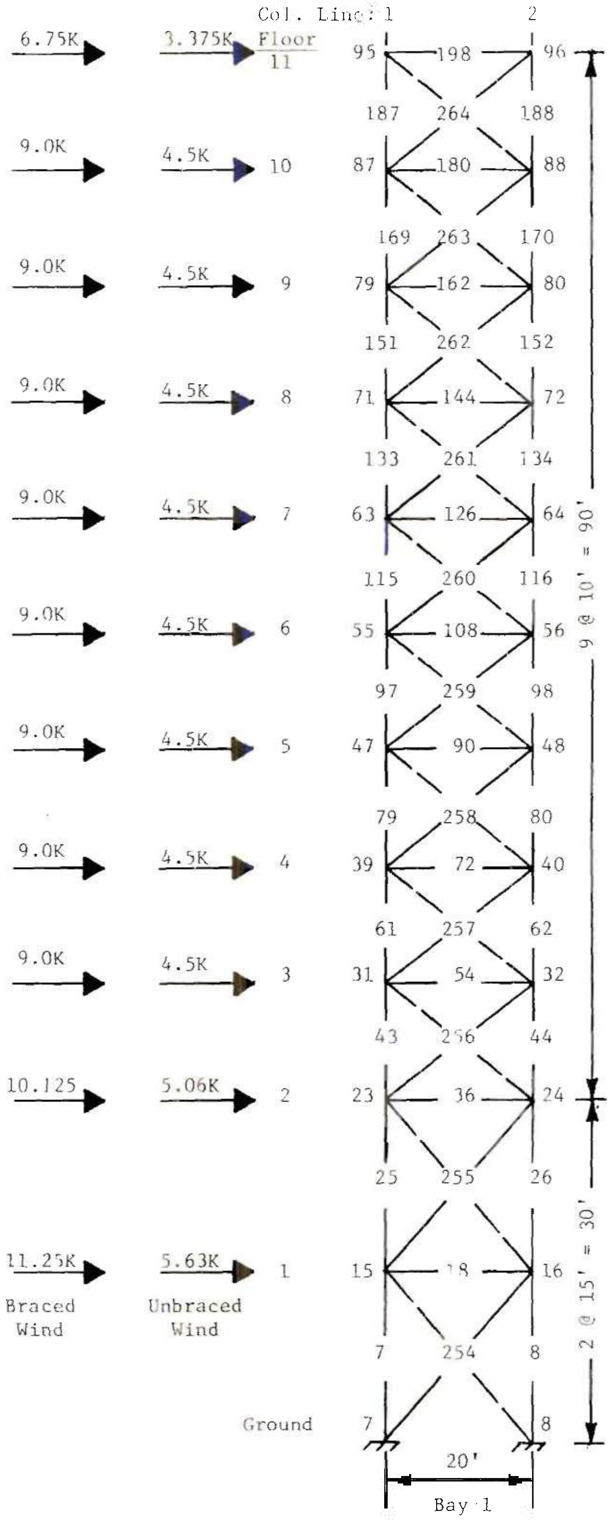


Figure 5.5 Frame B, Column Line C

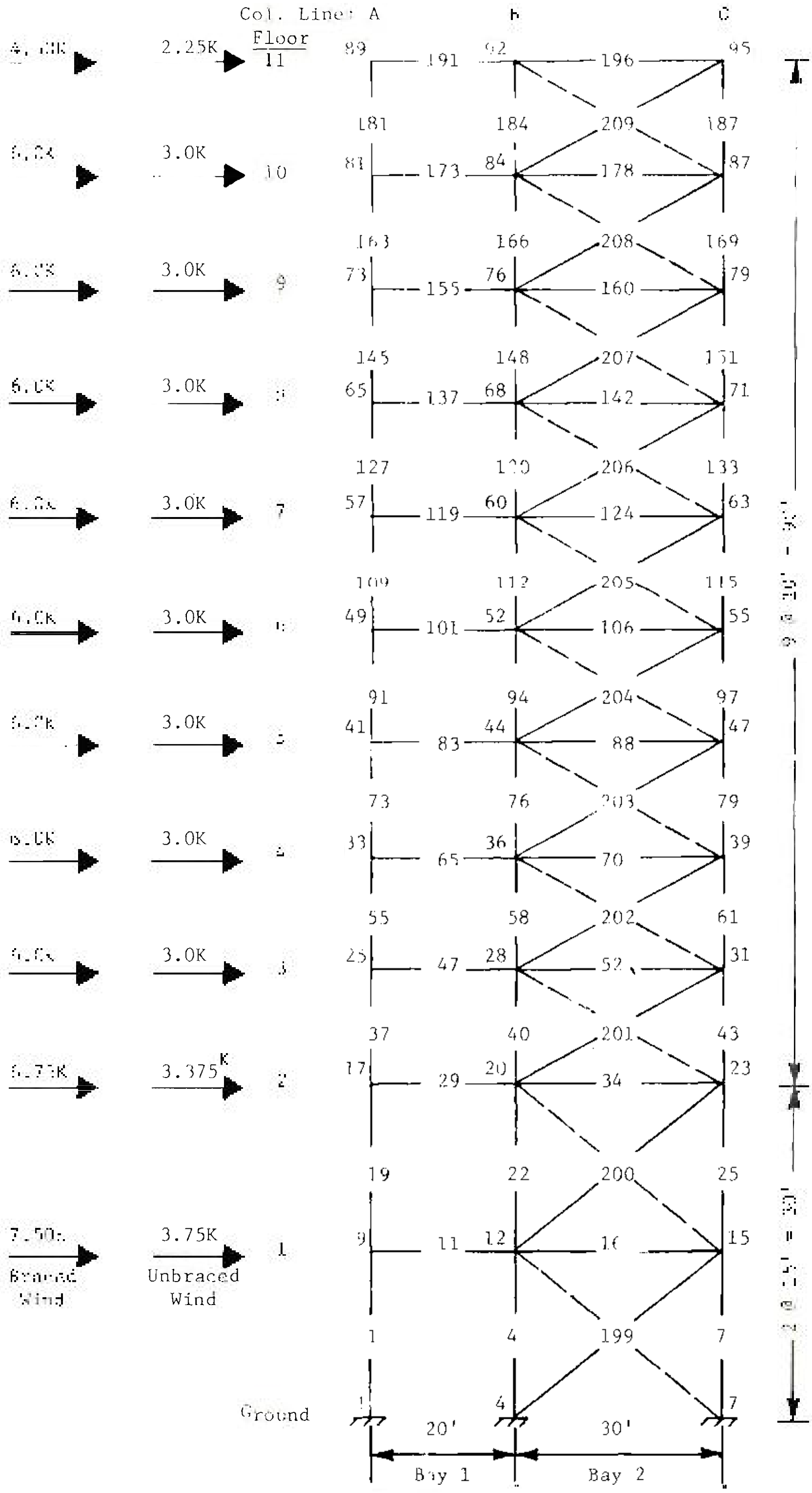


Figure 5.6 Frame B, Column Line 1

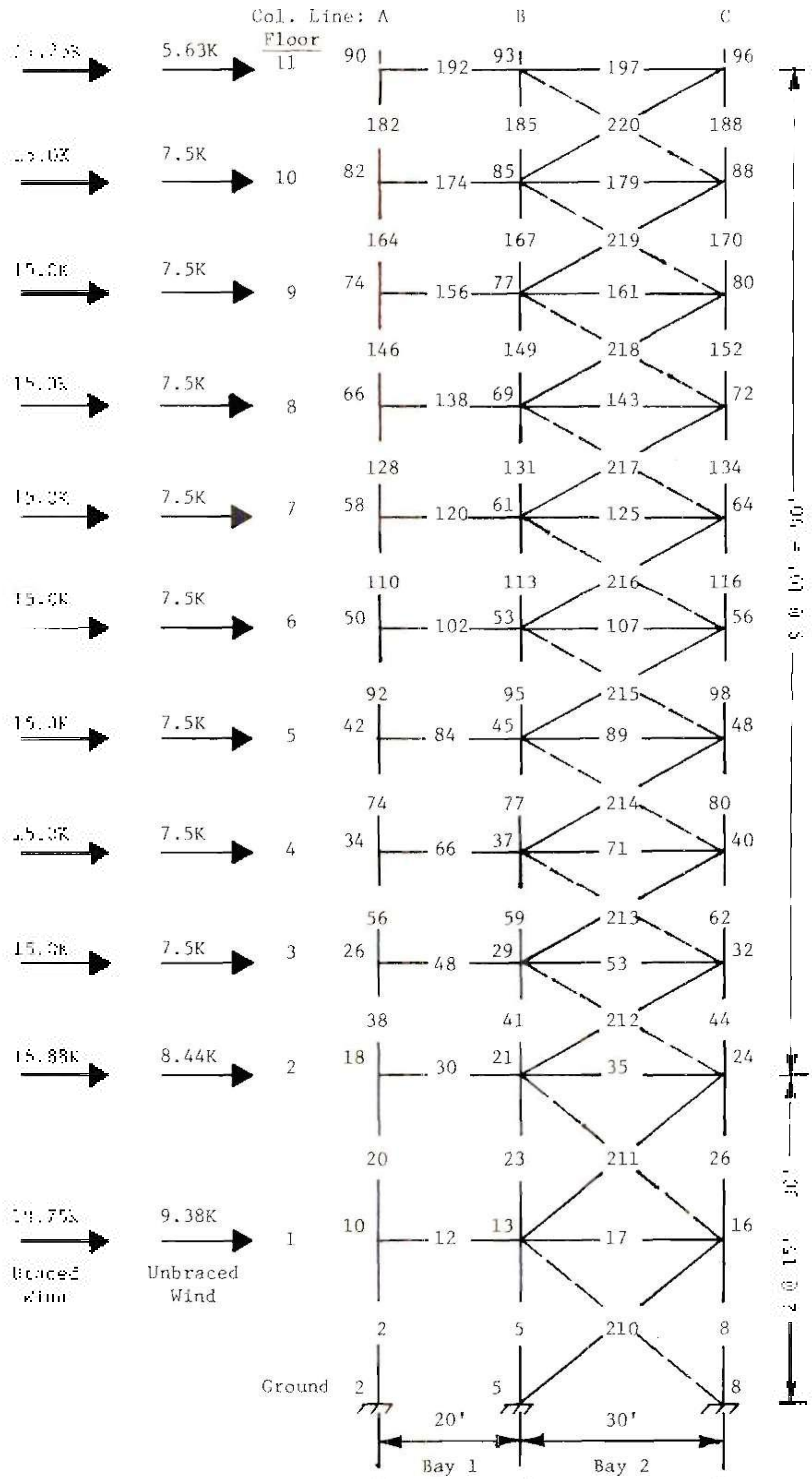


Figure 5.7 Frame B, Column Line 2

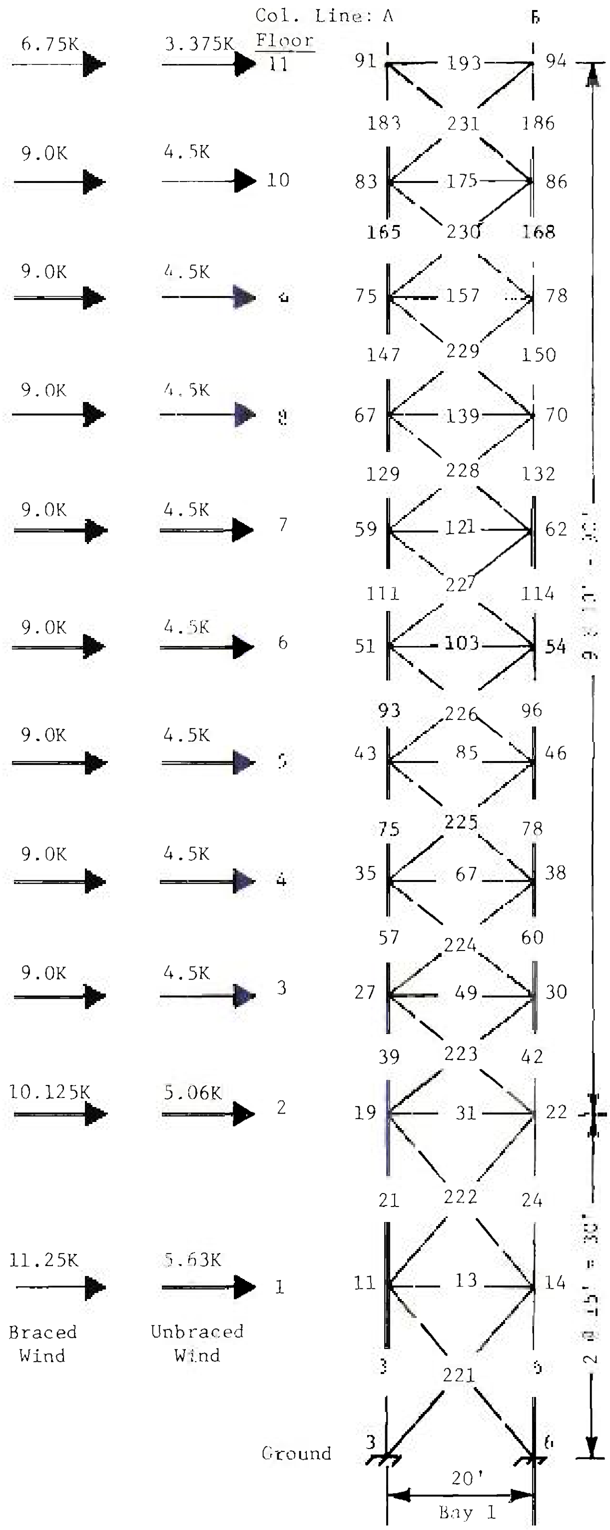


Figure 5.8 Frame B, Column Line 3

5.4 List of Example Problems with Additional Design Data

Eighteen design examples are presented for the purpose of displaying actual results. Important results and parameters from these examples are summarized in Tables 5.11 and 5.12. Of the eighteen examples, ten are presented in detail with initial and final section sizes in Tables 5.1 to 5.10. For these ten examples, graphs of displacement versus number of member size changes are plotted in Figs. 5.9 to 5.18. Average floor displacement versus floor number is plotted in Figs. 5.19 to 5.33.

The following design input data was identical for all examples except where noted (further discussion of these parameters can be found in the input descriptions in Appendix A).

1. The column section table consisted of 48 economy sections (ICOL = 48).
2. The beam section table consisted of 38 economy sections (IBEAM = 38).
3. The bracing section table consisted of 20 sections (IBR = 20).

   (Note: These section tables are in Appendix B. Also, only an unequal leg double angle bracing section table was used.)
4. A maximum of 250 member size changes were permitted between successive 'exact' analyses (IUPD = 250).
5. One member size change was permitted for each displacement constraint prior to the next virtual work cycle (MEMCHG = 1).
6. All examples had two displacement constraints except for the

unbraced plane frame, Examples A1, A2, and A3, which had three (NDC = 2 or 3).

7. The unit load magnification factor was 100.00 (FACT = 100.00).

8. The displacement constraint tolerance of convergence was 0.00 (TOL = 0.00).

9. Initial analysis type parameter indicated a stiffness analysis (IFLAG = 0). (Note: exact stiffness analysis rather than kinematically condensed analysis.)

10. Final analysis type parameter indicated a stiffness analysis (IFLAG1 = 0). (Note: exact stiffness analysis rather than kinematically condensed analysis.)

11. Maximum number of analyses (both stiffness and kinematically reduced) allowed was 10 (ITOTAL = 10).

12. The number of kinematically condensed analyses permitted per stiffness analysis was 0 (IEXACT = 0). (Note: No kinematically condensed analyses were performed in these eighteen examples.)

13. Only translational displacement constraints were imposed (IDTYP(I) = 1). For Frame A, these constraint values were computed as h/500, where h was the vertical height from the ground up to the constraint joint. At the top, floor 24, this was 6.91 inches (DC(I) = 6.91). At floor 18 it was 4.90 inches (DC(I) = 4.90). For Frame B constraint values were computed as h/400. At the top floor this was 3.60 inches (DC(I) = 3.60).

Note that although not specified in the input, units were assumed to be kips and inches. The system requires a consistent set of units be used.

The example problems are as follows. Note that the parameter EP (refer to input description in Appendix A for a more detailed description) is the initial error term specified.

1. Frame A - Plane Frame

   a. Group 1 - Unbraced frame, Wind in both directions, each applied separately.

      1. Example A1 - EP = 25%

      2. Example A2 - EP = 10%

      3. Example A3 - EP = 0%

   b. Group 2 - Tension brace in Bay 3 only for Wind from left.

      1. Example AWL1 - EP = 25%

      2. Example AWL2 - EP = 10%

      3. Example AWL3 - EP = 0%

   c. Group 3 - Tension brace in Bay 3 only for Wind from right.

      1. Example AWR1 - EP = 25%

      2. Example AWR2 - EP = 10%

      3. Example AWR3 - EP = 0%

2. Frame B - Space frame

   a. Group 4 - Unbraced frame, Wind applied separately in positive X-and Z-directions.

      1. Example B1 - EP = 25%

      2. Example B2 - EP = 10%

      3. Example B3 - EP = 0%

   b. Group 5 - Braced frame, Wind applied in positive X-direction, tension braces only in Bay 2 of column lines A and B, and in Bay 1 of column line C; Wind applied in positive Z-direction, tension braces only in Bay 2 of column lines 1 and 2, and in Bay 1 of column line 3.

1. Example BW1 - EP = 25%
2. Example BW2 - EP = 10%
3. Example BW3 - EP = 0%

c. Group 6 - Braced frame, Wind applied in negative X-direction, tension braces in same Bays as in Group 5;Wind applied in negative Z-direction, tension braces in same Bays as in Group 5.

1. Example BRW1 - EP = 25%
2. Example BRW2 - EP = 10%
3. Example BRW3 - EP = 0%

5.5 General Discussion of Results

The preliminary structure, before stiffness design, is assumed to be both stable and of adequate strength to safely resist applied external loads. For the examples presented here, ICES STRUDL II (7,8) was used to estimate initial sizes to meet strength requirements. However, some members were reduced in size and loads increased to force larger deflections so that results from the stiffness design method could be clearly shown. Because a final strength design was not actually completed and loads varied from example to example, not all examples for each frame type may be compared. Specifically, braced and unbraced problems have no basis for comparison. Only those examples with identical starting member sizes permit a valid comparison of results. The object of this Section is to show how the computer design system functions, what type of results it can produce, how certain input parameters affect design, and to provide the user some experience so he might achieve the best possible results when using the system.

The stiffness design method is not intended to be a true final design, but a useful tool in accomplishing that goal. It indicates those areas within the frame which are most sensitive to the chosen displacement constraints. Odd or unusual combinations of member changes may take place in the course of a design sequence. For instance, one particular member may change many times, while an adjacent member might not change at all. From the point of view of fabrication and erection costs, this can be very inefficient, even though in terms of overall weight and material cost, it is very economical. Obviously, a trade off is necessary. An engineer, seeing where member size increases occur, creates a more even distribution of member changes that not only satisfy the constraints, but is also better suited to an overall cost-efficient design.

Now, the critical displacement by which one can judge stiffness design efficiency is the last deflection which satisfies its imposed displacement constraint. After all other constraints are met, no additional member size changes are associated with them. But, since each member increase affects all displacements, members changed to reduce the final deflection also cause the remaining deflections to decrease.

It was found that the virtual work solution for displacements are smaller than actual displacements as would be computed by an exact stiffness analysis. This was due to the assumptions made for the virtual work solution, whereby shear and torsion effects were neglected, and axial forces and bending moments were assumed to remain constant for member size changes during the gradient search cycling. Therefore, it was possible to have the virtual work deflection satisfy the constraint, while the exact displacement was excessive. Thus, more reduction based on virtual work displacements may be necessary so that the exact displacements can meet the imposed deflection constraints (see Figs. 5.9 to 5.18).

To compensate for this problem, error terms were incorporated into the design process. An initial error term was input as a percentage of the initial exact displacements computed at constraint points. The initial computed exact displacements and error terms are added such that the absolute value of each deflection corresponding to a displacement constraint is increased. This forces more reduction to occur in the virtual work phase of the program, which is the fastest in terms displacement computations on the computer.

The proper selection of an initial error term may cause enough displacement decrease during the first virtual work gradient search stiffness design cycle, so that when an exact displacement check is next made (i.e., after the second exact analysis), deflections are below, but very close to the allowable limit. Of the eighteen examples discussed, only the braced frame problems with initial error terms of 25% and 10% (Examples AWL1, AWL2, AWR1, AWR2, BW1, BW2, BRW1, and BRW2) required only two exact analyses.

Although an initial error term may be chosen such that only two exact analyses (i.e., one to begin the process and one to check at the end) are required, this may not lead to the least cost frame design. A clear indication of this possible situation is shown by comparing in Table 5.12 Example BW1 (initial error term = 25%) which requires two analyses and has a final cost of $41,306.88, with Example BW3 (initial error term = 0%) which requires three analyses and has a final cost of $40,219.49.

Should more than a single gradient search process become necessary as in the case of Example BW3 (i.e., more than two exact analyses required), error terms in succeeding cycles are computed as the difference between the virtual work solution for displacements from the previous gradient search cycle and the exact displacements computed by the most recent exact analysis. These error terms are added as before.

The ultimate goal is to choose an initial error term which leads to a good final design for stiffness which satisfies the imposed displacement constraints, while at the same time reduces the amount of computer time required.

Note that no kinematic reduction analyses were used in any example. A discussion of why, along with a presentation of some results are found in Section 5.8.

Now, Figs. 5.9 to 5.18 show graphs of number of member changes versus displacement for certain selected examples. The horizontal lines on these graphs indicate the constraint values which are imposed at designated points in the structure. The initial displacements at the far left of each graph are determined from the first exact analysis of the structure under study, and all modifications to this value, during the gradient search cycle, were computed by virtual work calculations based on member size increases. In all cases, the plots show decreasing displacement at constraint locations due to a number of member size increases. The abrupt jumps in each line denote that an exact analysis is performed at that point because the approximate virtual work displacements plus an error term, are less than the constraint values for all constraints in the structure.

Notice that in some cases, there is a substantial change between the virtual work displacements at the end of a cycle and the exact displacements which are calculated by the next exact analysis. This difference was used as the compensating error term during the next gradient search cycle if the exact deflections were still greater than the allowable. The amount by which the virtual work solution was found to be below the constraint value was dependent upon the initial error term. This user input parameter has an important effect on program efficiency as will be shown later.

Though not obvious, notice that as the design process continues and more members are changed, the plots tend to flatten out, becoming slightly more horizontal. This is a consequence of the fact that members chosen to be increased have less effect on displacement than members picked previously. Those members with the largest influence are selected first, hence the beginning portion of the curve was generally steeper. This phenomenon also reflected the changing of internal member forces as members were increased in size. Recall that within a gradient search cycle, member forces were taken as constants. However, upon reanalyzing after many member size changes, forces may change significantly. Another major effect of assuming constant forces is the divergence of virtual work and exact displacements. This was compensated for by the use of the error term.

Figs. 5.19 to 5.33 are graphs of average story deflection as it varies with respect to floor number. The floor number is, in turn, directly related to vertical distance h, from the ground up to the floor in question. For Frame A, a plane frame, the horizontal displacement of joints on each floor were used to arrive at an average floor displacement. The same procedure was followed for Frame B, a space frame, except that this frame was divided up into the two orthogonal X and Z-directions, with each direction plotted separately. A deflection limit of h/400 was imposed on Frame B. The limiting displacement for Frame A was h/500. Note that it may not be concluded that since the average floor deflection satisfies the imposed constraint, every individual joint will satisfy also. This is primarily due to the fact that the floor may rotate.

The stiffness design system is highly efficient from a computer cost point of view. None of the eighteen examples (Tables 5.11 and 5.12) required more than five CPU minutes on an IBM 370/158. For example, consider the space frame Example B3 which had 198 members, 96 joints, 528 degrees of freedom and two displacement constraints. There were four separate loading conditions (both P and Q loads combined) for each exact stiffness analysis of the structure of which four were performed. There were between 45 and 90 virtual work cycles completed following each of the first three stiffness analyses, producing 331 member size increases.

All of the above was executed in less than four CPU minutes. At commercial rates of \$360 per CPU hour, four CPU minutes would cost about \$24. Now, compare this computer cost with the average cost increase of a typical member. Consider the additional cost resulting from an increase of a single member by one size in the appropriate section table. Suppose a member is 20 feet long, its cost factor is \$0.20 per pound of steel and the size increase produces a weight increase of five pounds per foot of length, which would be typical for a member in Example B3. The additional material cost for this member alone is \$20. Compare this value with the \$24 computer cost for the <u>total</u> design.

Therefore, one may conclude that minor variances in computer time for all examples (see Tables 5.11 and 5.12) are insignificant when compared to the material cost increases. The best overall design is that design which causes the smallest increase in structure weight while satisfying the displacement constraints.

5.6 Frame A Results - Details

5.6.1 Group 1 Results

Frame A is a plane frame shown in Fig. 5.1. Unbraced Examples A1, A2, and A3 all have identical preliminary member sizes. They compose Group 1 examples identified in Table 5.11. Both initial and final member sizes for these problems are found in Tables 5.1 and 5.2. Displacement constraints 1, 2, and 3 exist at joints 97, 100, and 69 with constraint values of 6.91, 6.91 and 4.90 inches respectively, which were derived from a limiting deflection constraint of h/500. Constraint positions 1 and 2 were on floor 24 with position 3 on floor 18. The unbraced Frame A was subject to loading conditions for wind from the left, associated with constraints 1 and 3, and wind from the right, associated with constraint 2. Because deflections at constraint locations 1 and 2 were so nearly identical, only constraint 1 was plotted in Figs. 5.9 to 5.11.

Consider now Figs. 5.9, 5.10 and 5.11 showing graphs relating the relevant displacements in the Group 1 examples. Table 5.11 summarizes their results. In all three examples, displacement constraint 1 (6.91 in.) was satisfied after about 45 member size increases. Constraint 3 (4.90 in.) was satisfied after about 50 size changes. However, note that in Example A1 (Fig. 5.9), a total of 90 member size changes were required before the next exact analysis was performed. This was due to an initial error term (EP) of 25% (i.e., the computed virtual work displacement plus 25% of the initial exact displacement had to satisfy the constraint). In Example A2 (Fig. 5.10), only 64 size changes occurred before the next exact analysis since EP was only 10%. Finally, in Example A3 (Fig. 5.11), the next analysis was performed after only 50 member size changes which first satisfied the constraints since EP was 0%.

The largest difference between the virtual work displacements and exact displacements are shown by Example A1. However, note that although there is a sizable difference, the exact displacements exceed the constraint values by only a small amount, after the second exact analysis. Example A2 produces a smaller difference, but the exact displacements after the second exact analysis were further above the constraint values. Example A3 shows a still smaller difference, but the exact deflections after the second exact analysis were even further above the allowable. The difference between the last virtual work deflections and the next exact deflections become the error terms for the following gradient search cycle.

Although the effect of the initial error term was most evident on the first gradient search cycle, its effects were also felt on subsequent analyses. Notice in Example A1, despite the fact that the exact displacement computed by the second analysis exceeded the constraint by only a small margin, the large error term accumulated for the second gradient search was large enough so that when the design concluded, the final displacement was more than 8% below the maximum allowable level. Example A3 exhibits the fact that an initial error term of 0% produced an ensuing sequence of error terms which were small, thus forcing one more analysis to be executed. Displacements in this case were still 8% below the constraint value. Finally, Example A2, with an initial error term of 10%, had a critical deflection which was only 3.5% below the constraint value. In terms of critical deflections, Example A2 was the best design.

Figs. 5.19, 5.20 and 5.21 graphically show the average floor displacements for these three examples both before and after stiffness design, and compares them to the deflection criterion of h/500. It is observed that the initial structure satisfied the deflection criterion for the first five floors. From this floor to the top floor, displacements were unsatisfactory. The final average floor displacements show that the final designs successfully reduced all story deflections to less than h/500. Keep in mind, however, that constraints were only dictated at floors 18 and 24, with no restrictions imposed at any other floors. It is interesting to note how the intermediate constraint point at floor 18, in addition to the constraint at the top, floor 24, produced a satisfactory design throughout in terms of satisfying a design displacement criterion of h/500. It will be shown that the space frame examples, without the intermediate constraint point, do not always behave in this manner.

In Group 1, the best design (i.e, lightest weight) and, therefore, the least costly structure (all members had identical cost factors, so cost was directly related to weight), was Example A2, with an initial error term of 10%. It was 2.1% lighter than Example A1, and 1.2% lighter than Example A3. This was translated into material cost savings of $1,404 and $826 over Examples A1 and A3, respectively (see Table 5.11).

5.6.2 Group 2 and 3 Results

Example AWL2 contained tension braces for wind load from the left, while Example AWR2 contained tension braces for wind from the right. Both examples have identical initial member properties and geometrical configurations and are the same except for the brace directions. Example AWL2 had displacement constraints 1 and 2 at joints 97 and 69 with values of 6.91 and 4.90 inches, respectively. Example AWR2 had corresponding displacement constraints on the right side of the structure, 1 and 2 at joints 100 and 72, with identical values of 6.91 and 4.90 inches respectively.

Figs. 5.12 and 5.13, and Table 5.11 show dispacements at constraint locations as a function of member size changes for both braced examples. These two Examples (AWL2 and AWR2 with initial error terms of 10%) were chosen, rather than AWR1, AWL1, etc., because they produced reasonable results, they were typical for Groups 2 and 3, and more importantly, they were associated with an initial error term of 10% which is recommended for structures of the type considered in this study.

Member sizes, both preliminary and final are listed in Tables 5.3, 5.4, and 5.5. Neither of the displacement constraints were exceeded by a very large amount and thus a solution was obtained rapidly in one gradient search cycle and after two exact analyses. The initial deflections were found to be slightly higher in Example AWL2 than AWR2, but the difference was small. Displacement constraint 1 in both cases was satisfied after five or fewer member changes. Displacement constraint 2 required just less than 20 size changes in Example AWR2, but required about 25 size changes in Example AWL2. Due to the initial error term of 10%, members continued to be increased beyond that required by the virtual work displacement calculation alone as shown in Figs. 5.12 and 5.13. When the gradient search was complete, Examples AWL2 and AWR2 required 62 and 50 member size changes, respectively, and upon completion of the second exact analysis, it was determined that the critical displacement (associated with constraint 2) in each case was less than the constraint value by only 2.2%, a very efficient design in terms of critical deflection. Note that only two analyses were performed in order to satisfy the constraints, as opposed to at least three in the unbraced cases. However, the initial displacements in the braced cases were much less than the unbraced cases. Actually, the braced and unbraced examples may not properly be compared because a common starting point for these examples was not employed.

In Figs. 5.22 and 5.23 the overall initial and final average floor displacements are compared with the h/500 design criterion. The first seven initial floor displacements for both examples satisfied the constaint values and all final deflections were at satisfactory levels. Recall that constraints were imposed only at floor 18 and 24 as was the situation with the unbraced structures.

Now, consider the Group 2 examples (Table 5.11) only. The lightest weight and least costly structure was Example AWL3. This example was 0.15% lighter and $88 less expensive than Example AWL2, as well as 0.64% lighter and $375 less expensive than Example AWL1.

Next, consider Group 3 examples (Table 5.11) only. The lightest weight and least costly structure was Example AWR3. This example was 0.18% lighter and $104 less expensive than Example AWR2, as well as 1.4% lighter and $800 less expensive than Example AWR1.

5.6.3 Summary of Frame A Results

Frame A examples were divided into Groups 1, 2, and 3, with Group 1 unbraced, and Groups 2 and 3 braced frames. All results are summarized in Table 5.11. Only tension braces were included in the Group 2 and 3 structures.

From Group 1, Example A2 with an initial error term of 10% produced the lightest weight and least expensive final design, as well as producing the best design in terms of critical deflections.

From Group 2, Example AWL3 with an initial error term of 0% resulted in the lightest and least expensive design, and also was the best in terms of critical deflections. Observe, however, that Example AWL2 with an initial error of 10%, produced a nearly identical design.

From Group 3, Example AWR3 with an initial error term of 0% resulted in the lightest and least expensive design, and also was the best as far as critical deflections were concerned. Note that Example AWR2 with an initial error term of 10% also produced a nearly identical design.

It is interesting to note where member size changes occurred in each of the Frame A examples. By a large margin, beams were chosen as the most sensitive to the displacement constraints (more beam size changes than columns or braces) in both braced and unbraced cases. For Example A2, beam changes were concentrated toward the lower half of Bay 2 and the middle of Bay 1. Several column changes were made toward the top of column lines B and D, and a few relatively scattered beam increases are found in Bay 3.

Example AWL2 for wind from the left and Example AWR2 for wind from the right are extremely similar in their member change patterns. The only braces to increase were those in Bay 3 between floors 6 and 16, one increase for each brace. The same columns at the top two stories of column line B increased once in each example, as well as several beams in Bay 3. Example AWL2 showed one additional beam change in Bay 3. The remainder of the member changes in the two examples were concentrated in Bay 2. Most of these were increased by two sizes with Example AWL2 showing several additional increases.

The close resemblance between Examples AWL2 and AWR2 implies that, at least for this plane frame, only one wind load case needed to be considered, since the opposite wind will produce like results, not only in beam and column sizes, but brace sizes as well. For the actual structure, tension braces were necessary to resist lateral loads in each direction separately, and these results showed that the same size braces may be used for wind in each direction. In practical engineering applications, this is usually the case due to fabrication and erection economy.

Generally, one may draw several conclusions from studying all Frame A examples. First, it was shown that an intermediate constraint imposed at floor 18, in addition to the constraint at the top floor 24, produced average floor displacements below the h/500 design criterion for all floors. Also, the initial error term of 25% resulted in the heaviest final design within each individual Group, while 10% resulted in the least weight design for the unbraced cases, and 0% resulted in the least weight design for the braced cases. For the braced cases, however, 0% and 10% initial error terms produced nearly identical results.

The computer time necessary to complete a design was discovered to be insignificant in comparison with the material weight increases as discussed in previous sections of this chapter.

Detailed results, including initial and final member sizes are shown for Examples A1, A2, A3, AWL2 and AWR2 in Tables 5.1 to 5.5. Figs. 5.9 to 5.13 show graphs of number of member changes versus displacement, and Figs. 5.19 to 5.23 show graphs of average floor displacement versus floor number for each of these examples.

5.7 Frame B Results - Details

5.7.1 Group 4 Results

The floor plan for Frame B, a space frame, is illustrated in Fig. 5.2. An identical plan exists for all 11 floors in the frame. Each of the column lines are shown in Figs. 5.3 to 5.8. Unbraced frame Examples B1, B2, and B3 all have identical preliminary member sizes. They compose Group 4 examples identified in Table 5.12. Both initial and final member sizes for these Examples are found in Tables 5.6 and 5.7. Displacement constraints 1 and 2 are specified on the top floor, at joints 92 and 90 in the X and Z-directions, respectively. Constraint limits are 3.60 inches for both directions, based on a deflection criterion used for this frame of h/400. Frame B is subject to wind loads in the positive X-direction associated with constraint 1, and positive Z-direction, associated with constraint 2. These directions correspond to the X and Z-directions shown in Fig. 5.2.

Figs. 5.14, 5.15 and 5.16 are graphs relating member changes and displacements at constraint locations for Group 4 examples. In a similar way as Group 1 examples, the effects of differences in initial error terms for each individual example is clearly evident. At the conclusion of the <u>first</u> gradient search cycle, Example B1 with an initial error of 25%, required 153 member size increases, while Example B2 with an initial error of 10% required 105 member size increases, and Example B3 with an initial error term of 0% required only 79 member size increases. The 25% term resulted in the largest difference between the last virtual work displacement from the gradient search loop and the following displacement from the exact stiffness analysis, while the 10% term had the next smaller difference, and finally, the 0% term had the smallest difference. This was precisely the same behavior found in the unbraced Frame A Examples of Group 1.

Example B3 required an additional analysis (a fourth analysis) to attain the final design, and overdesigned the critical displacement by almost 12%. Although it takes one less analysis (only three)than Example B3, Example B1 also overdesigned the critical displacement by about 11%. Finally, Example B2, which required three analyses as did Example B1, came within 0.3% of the constraint value for the critical deflection. Therefore, in terms of critical deflections, the best design in this group was Example B2.

These examples were quite different in terms of the number of member size increases necessary to achieve a final design. Example B2, sing an initial error term of 10%, required 90 fewer member size increases than either of the other two examples.

Figs. 5.24 to 5.29 show the average floor displacements in both the X and Z-directions for the Group 4 examples, and compare the initial and final floor displacements to the design constraint of h/400. For all six cases, the initial floor displacements exceeded the limit by a large margin, becoming progressively worse toward the top of the structure. The X-direction floor displacements of the Example B1 final design (Fig. 5.24) were reduced such that they were below the constraint at the top, but still slightly over the limit toward the bottom. The Z-direction floor displacements for this Example (Fig. 5.25) satisfied the constraint everywhere. In Example B3, both the X and Z-directions exhibited nearly the identical floor displacements as described for Example B1. In Example B2, the X-direction average floor displacement nowhere satisfied the h/400 criterion. However, at the top, the discrepancy was negligible. The Z-direction floor displacements satisfied the constraint only at the top story, but exceeded it at all other stories.

It must be noted that although the average floor displacements may be greater than the allowable, the individual constrained joints satisfy the displacement constraints. The displacement fields shown for unbraced Frame B had one marked difference from those of Frame A. The Frame B examples had displacement constraints imposed only at the top, as opposed to Frame A which had an additional constraint three quarters of the way up the building. The extra constraint in Frame A was the major reason why Frame A average floor displacements were entirely within the allowable deflection envelope, while Frame B displacements were only at a satisfactory level at the top. It is, therefore, recommended that at least one displacement constraint be located at approximately mid-height of the structure for future design.

In Group 4, the best design (i.e., the lightest weight) and consequently, the least expensive structure (all members had identical cost factors, so cost was directly related to weight), was Example B2, with an initial error term of 10%. It was 4.1% lighter than Example B1, and 4.6% lighter than Example B3. This was translated into material cost savings of \$2,325 and \$2,158 over Examples B1 and B3, respectively (see Table 5.12).

5.7.2 Group 5 and 6 Results

Figs. 5.17 and 5.18 show plots of member size increases versus displacement at constraint locations for braced Frame B Examples BW2 and BRW2, both with initial error terms of 10%. These were selected, rather than BW1, BRW1, etc., because they produced reasonable results, they were typical for Groups 5 and 6, and more importantly, they were associated with an initial error term of 10% which is recommended for structures of the type considered herein.

Examples BW2 and BRW2 had the same initial member sizes. These, as well as final design member sizes, are listed in Table 5.8, 5.9 and 5.10. Table 5.12 summarizes other design results for these examples.

Example BW2 was subjected to wind loads in the positive X-direction, associated with constraint 1 at joint 92,(on the top floor) and wind in the positive Z-direction, associated with constraint 2 at joint 90 (also on the top floor). Example BRW2 was subjected to negative X-direction wind, associated with constraint 1 at joint 94, and negative Z-direction wind associated with constraint 2 at joint 96. As with other Frame B examples, the upper bound on deflection was 3.60 inches. Notice that the constraint locations for the positive and negative loads correspond to each other, and were located on opposite sides of the same column line. Tension-only braces were used for wind in each direction.

Fig. 5.17 shows that initial displacements at constraint locations in Example BW2 were within one tenth of an inch of each other. After 160 member size increases, the virtual work displacements with a 10% error term satisfied the constraints. The subsequent exact analysis showed that the exact displacements were within 3% of the constraint value. Example BRW2 in Fig. 5.18 showed a similar trend. It required 130 member size increases before the virtual work displacements plus the 10% error term satisfied the constraint value. One may conclude from Table 5.12, that, in terms of critical displacements, Examples BW2 and BRW2 were the best in their respective groups, each having critical displacements within about 3% of constraint values.

Overall, the final results showed a difference of only 9 member size changes between the two examples (BW2 and BRW2) to complete the design. Observe that only two exact analyses were performed in each case (the initial one and the final one).

Plots of average floor displacements in the X and Z-directions for both examples are shown in Figs. 5.30, 5.31, 5.32 and 5.33. In each case, the initial displacements did not satisfy the h/400 constraint anywhere. After stiffness design, the displacement constraints were satisfied at every floor level. Note also that the displacements at the top were farther below the limit than those at the bottom.

Now, consider Group 5 examples (Table 5.12) only. The lightest weight and least costly structure was Example BW3. This example was 0.50% lighter and $205 less expensive than Example BW2, as well as 2.7% lighter and $1087 less expensive than Example BW1.

Next consider Group 6 examples (Table 5.12) only. The lightest weight and least costly structure was Example BRW3. This example was 0.13% lighter and $54 less expensive than Example BRW2, as well as 2.2% lighter and $907 less expensive than Example BRW1.

5.7.3 Summary of Frame B Results

Frame B examples were divided into Groups 4, 5, and 6 with Group 4 unbraced, and Groups 5 and 6 braced frames. All results are summarized in Table 5.12. Only tension braces were included in the Group 5 and 6 structures.

From Group 4, Example B2 with an initial error term of 10% produced the lightest weight and least expensive final design, as well as producing the best design in terms of critical deflections.

From Group 5, Example BW3 with an initial error term of 0% resulted in the lightest and least expensive design. Observe, however, that Example BW2, with an initial error term of 10%, produced a nearly identical design, while being the best in terms of critical deflections.

From Group 6, Example BRW3 with an initial error term of 0% resulted in the lightest and least expensive design. Note that Example BRW2, with an initial term of 10%, also produced a nearly identical design, while being the best in terms of critical deflections.

It is interesting to note the locations and types of member size changes which occurred. Example B2, previously described as the least costly structure of the unbraced examples in Group 4, showed beams having somewhat more influence on deflection than columns. Beams in column lines B and 2 (Figs. 5.4 and 5.7) were locations where most member increases occurred, with slightly more changes toward the top of the building. In both instances, Bay 1, the shorter bay, had the most beam size increases. The intersection of column lines B and 2 (B-2) contained most of the column size increases, where column locations B-1 and A-2 had much fewer column size increases. Columns in column lines C and 3 were relatively free of size increases.

The braced Examples BW2 for wind from the left, and BRW2 for wind from the right, showed remarkably similar results, which imply that only one wind load case needed to be considered, as was the case with Frame A examples, and that diagonal braces required to resist lateral loads in both directions may be of equal size. In practical engineering applications, this is usually the case due to fabrication and erection economy.

Braces proved to be the member type having the most significant effect on displacement constraints in these Examples. Column lines B and 2 (Figs. 5.4 and 5.7) showed significantly more bracing member size increases than any other part of the structure. Also in column lines B and 2, beams experiencing size increases were located entirely in Bay 1. Column location B-2 exhibited column size increases almost entirely between floors 4 and 9. Column lines A, C, 1, and 3 were generally unaffected, although a few small increases did occur.

Generally, one may draw several conclusions from studying all Frame B examples. First, for the unbraced frames (B1, B2, B3) it was shown that due to the lack of an intermediate constraint (i.e., a constraint somewhere between the ground and the top floors) that average floor displacements in the middle of the frame exceeded the h/400 displacement constraint boundary. For the unbraced frames, this caused no problems. Also, the initial error term of 25% resulted in the heaviest, and therefore costliest final design within each individual Group, while 10% resulted in the least weight design for the unbraced cases, and 0% resulted in the least weight design for the braced cases. For the braced cases, however, 0% and 10% initial error terms produced nearly identical results.

The computer time necessary to complete a design was discovered to be insignificant in comparison with the material weight increases as discussed in previous sections of this chapter.

5.8 Kinematic Condensation Results

The kinematic condensation technique detailed in Chapter 3 was intended at the outset to reduce the number of unknowns and thereby reduce

Table 5.1. Column Sizes - Unbraced Frame A

Member Number	Story Number	Initial Sizes	Final Design Example A1	Final Design Example A2	Final Design Example A3
1	1	14WF314	14WF314	14WF314	14WF314
2	1	14WF426	14WF426	14WF426	14WF426
3	1	14WF500	14WF500	14WF500	14WF500
4	1	14WF370	14WF370	14WF370	14WF370
8	2	14WF314	14WF314	14WF314	14WF314
9	2	14WF426	14WF426	14WF426	14WF426
10	2	14WF500	14WF500	14WF500	14WF500
11	2	14WF370	14WF370	14WF370	14WF370
15	3	14WF287	14WF287	14WF287	14WF287
16	3	14WF370	14WF370	14WF370	14WF370
17	3	14WF426	14WF426	14WF426	14WF426
18	3	14WF342	14WF342	14WF342	14WF342
22	4	14WF287	14WF287	14WF287	14WF287
23	4	14WF370	14WF370	14WF370	14WF370
24	4	14WF426	14WF426	14WF426	14WF426
25	4	14WF342	14WF342	14WF342	14WF342
29	5	14WF246	14WF246	14WF246	14WF246
30	5	14WF314	14WF314	14WF314	14WF314
31	5	14WF370	14WF370	14WF370	14WF370
32	5	14WF314	14WF314	14WF314	14WF314
36	6	14WF246	14WF246	14WF246	14WF246
37	6	14WF314	14WF314	14WF314	14WF314
38	6	14WF370	14WF370	14WF370	14WF370
39	6	14WF314	14WF314	14WF314	14WF314
43	7	14WF228	14WF228	14WF228	14WF228
44	7	14WF264	14WF264	14WF264	14WF264
45	7	14WF314	14WF314	14WF314	14WF314
46	7	14WF287	14WF287	14WF287	14WF287
50	8	14WF228	14WF228	14WF228	14WF228
51	8	14WF264	14WF264	14WF264	14WF264
52	8	14WF314	14WF314	14WF314	14WF314
53	8	14WF287	14WF287	14WF287	14WF287
57	9	14WF202	14WF202	14WF202	14WF202
58	9	14WF219	14WF219	14WF219	14WF219
59	9	14WF264	14WF264	14WF264	14WF264
60	9	14WF246	14WF246	14WF246	14WF246
64	10	14WF202	14WF202	14WF202	14WF202
65	10	14WF219	14WF219	14WF219	14WF219
66	10	14WF264	14WF264	14WF264	14WF264
67	10	14WF246	14WF246	14WF246	14WF246

Table 5.1. Continued

Member Number	Story Number	Initial Sizes	Final Design Example A1	Final Design Example A2	Final Design Example A3
71	11	14WF176	14WF176	14WF176	14WF176
72	11	14WF184	14WF184	14WF184	14WF184
73	11	14WF219	14WF219	14WF219	14WF219
74	11	14WF219	14WF219	14WF219	14WF219
78	12	14WF176	14WF176	14WF176	14WF176
79	12	14WF184	14WF184	14WF184	14WF184
80	12	14WF219	14WF219	14WF219	14WF219
81	12	do	do	do	do
85	13	14WF150	14WF150	14WF150	14WF150
86	13	do	do	do	do
87	13	14WF193	14WF193	14WF193	14WF193
88	13	do	do	do	do
92	14	14WF150	14WF150	14WF150	14WF150
93	14	do	do	do	do
94	14	14WF193	14WF193	14WF193	14WF193
95	14	do	do	do	do
99	15	14WF127	14WF127	14WF127	14WF127
100	15	do	do	do	do
101	15	14WF158	14WF158	14WF158	14WF158
102	15	14WF167	14WF167	14WF167	14WF167
106	16	14WF127	14WF127	14WF127	14WF127
107	16	do	do	do	do
108	16	14WF158	14WF158	14WF158	14WF158
109	16	14WF167	14WF167	14WF167	14WF167
113	17	14WF111	14WF111	14WF111	14WF111
114	17	do	do	do	do
115	17	14WF136	14WF136	14WF136	14WF136
116	17	do	do	do	do
120	18	14WF111	14WF111	14WF111	14WF111
121	18	do	do	do	do
122	18	14WF136	14WF136	14WF136	14WF136
123	18	do	do	do	do
127	19	14WF84	14WF84	14WF84	14WF84
128	19	do	do	do	do
129	19	14WF111	14WF111	14WF111	14WF111
130	19	do	do	do	do
134	20	14WF84	14WF84	14WF84	14WF84
135	20	do	do	do	do
136	20	14WF111	14WF111	14WF111	14WF111
137	20	do	do	do	do

Table 5.1. Continued

Member Number	Story Number	Initial Sizes	Final Design Example A1	Final Design Example A2	Final Design Example A3
141	21	14WF61	14WF61	14WF61	14WF61
142	21	14WF53	14WF53	14WF53	14WF53
143	21	14WF74	14WF74	14WF74	14WF74
144	21	14WF78	14WF84	14WF84	14WF84
148	22	14WF61	14WF61	14WF61	14WF61
149	22	14WF53	14WF53	14WF53	14WF53
150	22	14WF74	14WF74	14WF74	14WF74
151	22	14WF78	14WF84	14WF84	14WF84
155	23	12WF40	12WF40	12WF40	12WF40
156	23	10WF39	14WF43	14WF43	14WF43
157	23	14WF48	14WF48	14WF48	14WF48
158	23	14WF53	14WF53	14WF53	14WF53
162	24	12WF40	12WF40	12WF40	12WF40
163	24	10WF39	12WF40	12WF40	12WF40
164	24	14WF48	14WF48	14WF48	14WF48
165	24	14WF53	14WF53	14WF53	14WF53

Table 5.2. Beam Sizes - Unbraced Frame A

Member Number	Story Number	Initial Sizes	Final Design Example A1	Final Design Example A2	Final Design Example A3
5	1	18WF45	18WF45	18WF45	18WF45
6	1	33WF118	33WF118	33WF118	36WF135
7	1	24WF68	24WF68	24WF68	24WF68
12	2	18WF45	18WF45	18WF45	18WF45
13	2	30WF116	36WF150	36WF135	36WF150
14	2	24WF68	24WF68	24WF68	24WF68
19	3	18WF45	18WF45	18WF45	18WF45
20	3	30WF108	36WF160	36WF160	36WF150
21	3	24WF68	24WF68	24WF68	24WF68
26	4	18WF45	18WF45	18WF45	18WF45
27	4	30WF99	36WF160	36WF160	36WF135
28	4	24WF68	24WF68	24WF68	24WF68
33	5	18WF45	18WF45	18WF45	18WF45
34	5	30WF99	36WF160	36WF160	36WF135
35	5	24WF68	24WF68	24WF68	24WF68
40	6	18WF45	18WF45	18WF45	18WF45
41	6	27WF94	36WF135	36WF160	36WF135
42	6	24WF68	24WF68	24WF68	24WF68
47	7	18WF45	21WF55	21WF55	21WF55
48	7	27WF84	30WF99	36WF135	33WF118
49	7	24WF68	24WF68	24WF68	27WF84
54	8	18WF45	24WF68	24WF68	24WF68
55	8	27WF84	30WF108	30WF108	30WF108
56	8	24WF68	33WF130	24WF68	27WF84
61	9	18WF45	27WF84	24WF76	24WF68
62	9	24WF76	36WF160	30WF99	30WF99
63	9	24WF68	24WF68	24WF68	27WF84
68	10	18WF45	27WF84	24WF68	27WF84
69	10	24WF68	27WF84	33WF118	36WF135
70	10	24WF68	24WF68	30WF99	27WF84
75	11	18WF45	30WF108	24WF68	27WF84
76	11	21WF62	24WF68	27WF84	27WF84
77	11	24WF68	24WF68	24WF68	24WF68
82	12	18WF45	24WF68	30WF108	27WF84
83	12	18WF50	24WF68	21WF55	24WF68
84	12	24WF68	30WF99	24WF68	24WF76
89	13	18WF45	24WF68	24WF68	36WF135
90	13	16WF40	18WF45	21WF55	18WF45
91	13	24WF68	36WF135	27WF84	27WF84
96	14	18WF45	21WF55	30WF99	27WF84
97	14	14WF30	16B31	16B31	16B31

Table 5.2. Continued

Member Number	Story Number	Initial Sizes	Final Design Example A1	Final Design Example A2	Final Design Example A3
98	14	24WF68	30WF99	30WF108	27WF84
103	15	18WF45	21WF55	27WF84	27WF84
104	15	16B26	16B26	16B26	16B26
105	15	24WF68	30WF99	24WF68	30WF99
110	16	18WF45	27WF84	24WF68	27WF84
111	16	16B26	16B26	16B26	16B26
112	16	21WF62	24WF68	27WF84	27WF84
117	17	18WF45	27WF84	24WF68	27WF84
118	17	16B26	16B26	16B26	16B26
119	17	21WF55	27WF84	21WF55	21WF55
124	18	18WF45	27WF84	24WF68	27WF84
125	18	16B26	16B26	16B26	16B26
126	18	21WF55	27WF84	21WF55	21WF55
131	19	16WF40	24WF68	24WF68	24WF68
132	19	16B26	16B26	16B26	16B26
133	19	21WF55	27WF84	21WF55	24WF76
138	20	16WF36	24WF68	18WF45	21WF55
139	20	14B22	16B26	16B26	16B26
140	20	21WF55	24WF68	24WF68	27WF94
145	21	16WF36	24WF76	18WF45	21WF55
146	21	14B17.2	16B26	16B26	16B26
147	21	21WF55	27WF84	24WF68	21WF62
152	22	16WF36	18WF45	16WF36	16WF36
153	22	14B17.2	16B26	16B31	14B17.2
154	22	21WF55	21WF55	21WF55	21WF55
159	23	16WF36	16WF36	16WF36	16WF36
160	23	14B17.2	14B17.2	14B17.2	14B17.2
161	23	21WF55	21WF55	21WF55	21WF55
166	24	16B26	16B26	16B26	16B26
167	24	12JR11.8	12JR11.8	12JR11.8	12JR11.8
168	24	18WF45	18WF45	18WF45	18WF45

Table 5.3 Column Sizes - Braced Frame A

Member Number	Story Number	Initial Sizes	Final Design Example AWL2	Final Design Example AWR2
1	1	14WF314	14WF314	14WF314
2	1	14WF314	14WF314	14WF314
3	1	14WF398	14WF398	14WF398
4	1	14WF398	14WF398	14WF398
8	2	14WF314	14WF314	14WF314
9	2	14WF314	14WF314	14WF314
10	2	14WF398	14WF398	14WF398
11	2	14WF398	14WF398	14WF398
15	3	14WF287	14WF287	14WF287
16	3	14WF287	14WF287	14WF287
17	3	14WF370	14WF370	14WF370
18	3	14WF370	14WF370	14WF370
22	4	14WF287	14WF287	14WF287
23	4	14WF287	14WF287	14WF287
24	4	14WF370	14WF370	14WF370
25	4	14WF370	14WF370	14WF370
29	5	14WF246	14WF246	14WF246
30	5	14WF246	14WF246	14WF246
31	5	14WF314	14WF314	14WF314
32	5	14WF314	14WF314	14WF314
36	6	14WF246	14WF246	14WF246
37	6	14WF246	14WF246	14WF246
38	6	14WF314	14WF314	14WF314
39	6	14WF314	14WF314	14WF314
43	7	14WF228	14WF228	14WF228
44	7	14WF219	14WF219	14WF219
45	7	14WF287	14WF287	14WF287
46	7	14WF287	14WF287	14WF287
50	8	14WF228	14WF228	14WF228
51	8	14WF219	14WF219	14WF219
52	8	14WF287	14WF287	14WF287
53	8	14WF287	14WF287	14WF287
57	9	14WF202	14WF202	14WF202
58	9	14WF202	14WF202	14WF202
59	9	14WF246	14WF246	14WF246
60	9	14WF246	14WF246	14WF246
64	10	14WF202	14WF202	14WF202
65	10	14WF202	14WF202	14WF202
66	10	14WF246	14WF246	14WF246
67	10	14WF246	14WF246	14WF246

Table 5.3. Continued

Member Number	Story Number	Initial Sizes	Final Design Example AWL2	Final Design Example AWR2
71	11	14WF176	14WF176	14WF176
72	11	14WF176	14WF176	14WF176
73	11	14WF219	14WF219	14WF219
74	11	14WF219	14WF219	14WF219
78	12	14WF176	14WF176	14WF176
79	12	14WF176	14WF176	14WF176
80	12	14WF219	14WF219	14WF219
81	12	14WF219	14WF219	14WF219
85	13	14WF150	14WF150	14WF150
86	13	14WF150	14WF150	14WF150
87	13	14WF193	14WF193	14WF193
88	13	14WF193	14WF193	14WF193
92	14	14WF150	14WF150	14WF150
93	14	14WF150	14WF150	14WF150
94	14	14WF193	14WF193	14WF193
95	14	14WF193	14WF193	14WF193
99	15	14WF127	14WF127	14WF127
100	15	14WF127	14WF127	14WF127
101	15	14WF158	14WF158	14WF158
102	15	14WF167	14WF167	14WF167
106	16	14WF127	14WF127	14WF127
107	16	14WF127	14WF127	14WF127
108	16	14WF158	14WF158	14WF158
109	16	14WF167	14WF167	14WF167
113	17	14WF111	14WF111	14WF111
114	17	14WF111	14WF111	14WF111
115	17	14WF136	14WF136	14WF136
116	17	14WF136	14WF136	14WF136
120	18	14WF111	14WF111	14WF111
121	18	14WF111	14WF111	14WF111
122	18	14WF136	14WF136	14WF136
123	18	14WF136	14WF136	14WF136
127	19	14WF84	14WF84	14WF84
128	19	14WF78	14WF78	14WF78
129	19	14WF111	14WF111	14WF111
130	19	14WF111	14WF111	14WF111
134	20	14WF84	14WF84	14WF84
135	20	14WF78	14WF78	14WF78
136	20	14WF111	14WF111	14WF111

Table 5.3. Continued

Member Number	Story Number	Initial Sizes	Final Design Example AWL2	Final Design Example AWR2
137	20	14WF111	14WF111	14WF111
141	21	14WF61	14WF61	14WF61
142	21	14WF53	14WF53	14WF53
143	21	14WF74	14WF74	14WF74
144	21	14WF78	14WF78	14WF78
148	22	14WF61	14WF61	14WF61
149	22	14WF53	14WF53	14WF53
150	22	14WF74	14WF74	14WF74
151	22	14WF78	14WF78	14WF78
155	23	12WF40	12WF40	12WF40
156	23	10WF39	12WF40	12WF40
157	23	14WF48	14WF48	14WF48
158	23	14WF53	14WF53	14WF53
162	24	12WF40	12WF40	12WF40
163	24	10WF39	12WF40	12WF40
164	24	14WF48	14WF48	14WF48
165	24	14WF53	14WF53	14WF53

Table 5.4 Beam Sizes - Braced Frame A

Member Number	Story Number	Initial Sizes	Final Design Example AWL2	Final Design Example AWR3
5	1	16WF36	16WF36	16WF36
6	1	14B17.2	14B17.2	14B17.2
7	1	21WF62	21WF62	21WF62
12	2	16WF36	16WF36	16WF36
13	2	14B17.2	16B26	14B17.2
14	2	21WF62	21WF62	21WF62
19	3	16WF36	16WF36	16WF36
20	3	14B17.2	16B26	16B26
21	3	21WF62	24WF68	24WF68
26	4	16WF36	16WF36	16WF36
27	4	14B17.2	16B26	16B26
28	4	21WF62	24WF68	24WF68
33	5	16WF36	16WF36	16WF36
34	5	14B17.2	16B26	16B26
35	5	21WF62	24WF68	21WF62
40	6	16WF36	16WF36	16WF36
41	6	14B17.2	16B26	16B26
42	6	21WF62	21WF62	21WF62
47	7	16WF36	16WF36	16WF36
48	7	14B17.2	16B26	16B26
49	7	21WF62	21WF62	21WF62
54	8	16WF36	16WF36	16WF36
55	8	14B17.2	16B26	16B26
56	8	21WF55	21WF55	21WF55
61	9	16WF36	16WF36	16WF36
62	9	14B17.2	16B26	16B26
63	9	21WF55	21WF55	21WF55
68	10	16WF36	16WF36	16WF36
69	10	14B17.2	16B26	16B26
70	10	21WF55	21WF55	21WF55
75	11	16WF36	16WF36	16WF36
76	11	14B17.2	16WF40	16B26
77	11	21WF55	21WF55	21WF55
82	12	16WF36	16WF36	16WF36
83	12	14B17.2	16B26	16B26
84	12	21WF55	21WF55	21WF55
89	13	16WF36	16WF36	16WF36
90	13	14B17.2	16B26	16B26
91	13	21WF55	21WF55	21WF55

Table 5.4. Continued

Member Number	Story Number	Initial Sizes	Final Design Example AWL2	Final Design Example AWR3
96	14	16WF36	16WF36	16WF36
97	14	14B17.2	16B26	16B26
98	14	21WF55	21WF55	21WF55
103	15	16WF36	16WF36	16WF36
104	15	14B17.2	16WF36	16B26
105	15	21WF55	21WF55	21WF55
110	16	16WF36	16WF36	16WF36
111	16	14B17.2	16B26	16B26
112	16	21WF55	21WF55	21WF55
117	17	16WF36	16WF36	16WF36
118	17	14B17.2	16B26	16B26
119	17	21WF55	21WF55	21WF55
124	18	16WF36	16WF36	16WF36
125	18	14B17.2	16B26	16B26
126	18	21WF55	21WF55	21WF55
131	19	16WF36	16WF36	16WF36
132	19	14B17.2	16B26	16B26
133	19	21WF55	21WF55	21WF55
138	20	16WF36	16WF36	16WF36
139	20	14B17.2	16B26	16B26
140	20	21WF55	21WF55	21WF55
145	21	16WF36	16WF36	16WF36
146	21	14B17.2	14B17.2	14B17.2
147	21	21WF55	21WF55	21WF55
152	22	16WF36	16WF36	16WF36
153	22	14B17.2	14B17.2	14B17.2
154	22	21WF55	21WF55	21WF55
159	23	16WF36	16WF36	16WF36
160	23	14B17.2	14B17.2	14B17.2
161	23	21WF55	21WF55	21WF55
166	24	16B26	16B26	16B26
167	24	12JR11.8	12JR11.8	12JR11.8
168	24	18WF45	18WF45	18WF45

Table 5.5 Brace Sizes - Braced Frame A

Member Number	Story Number	Initial Sizes	Final Design Example AWL2	Final Design Example AWR2
169	1	5UAN27.2	5UAN27.2	5UAN27.2
170	2	5UAN25.6	5UAN25.6	5UAN25.6
171	3	6UAN24.6	6UAN24.6	6UAN24.6
172	4	6UAN23.4	6UAN23.4	6UAN23.4
173	5	4UAN22.2	4UAN22.2	4UAN22.2
174	6	4UAN21.2	4UAN21.2	4UAN21.2
175	7	4UAN19.6	4UAN21.2	4UAN21.2
176	8	4UAN18.2	4UAN19.6	4UAN19.6
177	9	4UAN18.2	4UAN19.6	4UAN19.6
178	10	4UAN17.0	4UAN18.2	4UAN18.2
179	11	4UAN15.4	4UAN17.0	4UAN17.0
180	12	4UAN14.4	4UAN15.4	4UAN15.4
181	13	3UAN13.2	4UAN14.4	4UAN14.4
182	14	3UAN13.2	4UAN14.4	4UAN14.4
183	15	4UAN11.6	3UAN13.2	3UAN13.2
184	16	4UAN11.6	3UAN13.2	3UAN13.2
185	17	3UAN9.0	3UAN9.0	3UAN9.0
186	18	3UAN9.0	3UAN9.0	3UAN9.0
187	19	3UAN9.0	3UAN9.0	3UAN9.0
188	20	3UAN6.1	3UAN6.1	3UAN6.1
189	21	3UAN6.1	3UAN6.1	3UAN6.1
190	22	3UAN6.1	3UAN6.1	3UAN6.1
191	23	3UAN6.1	3UAN6.1	3UAN6.1
192	24	3UAN6.1	3UAN6.1	3UAN6.1

Table 5.6. Column Sizes - Unbraced Frame B

Member Number	Story Number	Initial Sizes	Final Design Example B1	Final Design Example B2	Final Design Example B3
1	1	14WF150	14WF150	14WF150	14WF150
2	1	do	14WF184	14WF158	14WF202
3	1	do	14WF150	14WF150	14WF150
4	1	do	do	do	14WF158
5	1	do	14WF184	14WF176	14WF202
6	1	do	14WF150	14WF150	14WF150
7	1	do	do	do	do
8	1	do	do	do	do
19	2	do	do	do	do
20	2	do	do	do	14WF158
21	2	do	do	do	14WF150
22	2	do	14WF158	do	14WF158
23	2	do	14WF211	14WF184	14WF219
24	2	do	14WF150	14WF150	14WF150
25	2	do	do	do	do
26	2	do	do	do	do
37	3	14WF119	14WF119	14WF119	14WF119
38	3	do	do	do	do
39	3	do	do	do	do
40	3	do	do	do	do
41	3	do	14WF150	14WF142	14WF167
42	3	do	14WF119	14WF119	14WF119
43	3	do	do	do	do
44	3	do	do	do	do
55	4	do	do	do	do
56	4	do	do	do	do
57	4	do	do	do	do
58	4	do	do	do	do
59	4	do	14WF142	14WF127	14WF142
60	4	do	14WF119	14WF119	14WF119
61	4	do	do	do	do
62	4	do	do	do	do
73	5	14WF84	14WF84	14WF84	14WF84
74	5	do	do	do	do
75	5	do	do	do	do
76	5	do	14WF111	14WF111	14WF111
77	5	do	14WF150	14WF127	14WF150
78	5	do	14WF84	14WF84	14WF84
79	5	do	do	do	do
80	5	do	do	do	do

Table 5.6. Continued

Member Number	Story Number	Initial Sizes	Final Design Example B1	Final Design Example B2	Final Design Example B3
91	6	14WF84	14WF84	14WF84	14WF84
92	6	do	do	do	do
93	6	do	do	do	do
94	6	do	14WF111	14WF111	14WF111
95	6	do	14WF142	14WF127	14WF142
96	6	do	14WF111	14WF84	14WF84
97	6	do	14WF84	do	do
98	6	do	do	do	do
109	7	14WF61	14WF61	14WF61	14WF61
110	7	do	14WF84	14WF78	14WF84
111	7	do	14WF61	14WF61	14WF61
112	7	do	14WF111	14WF84	14WF111
113	7	do	14WF142	14WF119	14WF142
114	7	do	12WF79	14WF61	14WF61
115	7	do	14WF61	do	do
116	7	do	do	do	do
127	8	do	do	do	do
128	8	do	do	do	14WF78
129	8	do	do	do	14WF61
130	8	do	14WF111	14WF84	12WF79
131	8	do	14WF142	14WF127	14WF127
132	8	do	12WF79	14WF61	12WF79
133	8	do	14WF61	do	14WF61
134	8	do	12WF79	do	do
145	9	12WF40	14WF43	14WF43	14WF43
146	9	do	14WF78	14WF78	14WF61
147	9	do	12WF40	12WF40	12WF40
148	9	do	14WF84	14WF84	12WF79
149	9	do	14WF111	14WF111	14WF127
150	9	do	12WF40	12WF40	12WF40
151	9	do	do	do	do
152	9	do	do	do	do
163	10	do	do	do	do
164	10	do	14WF48	14WF48	14WF61
165	10	do	12WF40	12WF40	12WF40
166	10	do	12WF58	12WF58	12WF58
167	10	do	14WF111	14WF119	14WF111
168	10	do	12WF40	12WF40	12WF40
169	10	do	do	do	do
170	10	do	do	do	do

Table 5.6. Continued

Member Number	Story Number	Initial Sizes	Final Design Example B1	Final Design Example B2	Final Design Example B3
181	11	8WF35	8WF35	8WF35	8WF35
182	11	do	14WF43	14WF43	14WF43
183	11	do	8WF35	8WF35	8WF35
184	11	do	do	do	do
185	11	do	14WF84	14WF84	14WF111
186	11	do	8WF35	8WF35	8WF35
187	11	do	do	do	do
188	11	do	do	do	do

Table 5.7. Beam Sizes - Unbraced Frame B

Member Number	Story Number	Initial Sizes	Final Design Example B1	Final Design Example B2	Final Design Example B3
9	1	21WF55	27WF84	24WF68	24WF68
10	1	do	21WF55	21WF55	21WF55
11	1	do	27WF84	27WF84	27WF84
12	1	do	33WF118	33WF118	36WF135
13	1	do	27WF84	27WF84	27WF84
14	1	do	33WF118	30WF99	33WF118
15	1	do	27WF84	27WF84	27WF84
16	1	do	21WF55	21WF55	21WF55
17	1	do	24WF68	do	do
18	1	do	27WF84	24WF68	24WF68
27	2	do	24WF68	21WF55	do
28	2	do	21WF55	do	21WF55
29	2	do	27WF84	24WF68	24WF68
30	2	do	30WF108	30WF99	33WF118
31	2	do	27WF84	27WF84	27WF84
32	2	do	30WF99	do	30WF108
33	2	do	27WF84	do	27WF84
34	2	do	21WF55	21WF55	21WF55
35	2	do	24WF68	24WF68	24WF68
36	2	do	do	do	do
45	3	do	21WF55	21WF55	21WF55
46	3	do	do	do	do
47	3	do	do	do	do
48	3	do	27WF84	27WF84	27WF84
49	3	do	24WF68	21WF55	21WF55
50	3	do	27WF84	27WF84	30WF99
51	3	do	24WF68	21WF55	24WF68
52	3	do	21WF55	do	21WF55
53	3	do	24WF68	do	24WF68
54	3	do	21WF55	do	21WF55
63	4	do	do	do	do
64	4	do	do	do	do
65	4	do	do	do	do
66	4	do	27WF84	27WF84	27WF84
67	4	do	24WF68	21WF55	24WF68
68	4	do	27WF84	27WF84	27WF84
69	4	do	24WF68	21WF55	24WF68
70	4	do	21WF55	do	21WF55
71	4	do	24WF68	do	do
72	4	do	21WF55	do	do

Table 5.7. Continued

Member Number	Story Number	Initial Sizes	Final Design Example B1	Final Design Example B2	Final Design Example B3
81	5	18WF45	18WF45	18WF45	18WF45
82	5	do	do	do	do
83	5	do	do	do	do
84	5	do	27WF84	27WF84	30WF99
85	5	do	21WF55	21WF55	21WF55
86	5	do	24WF68	24WF68	27WF84
87	5	do	do	21WF55	24WF68
88	5	do	18WF45	18WF45	18WF45
89	5	do	21WF55	do	do
90	5	do	18WF45	do	do
99	6	do	do	do	do
100	6	do	do	do	do
101	6	do	do	do	do
102	6	do	30WF99	27WF84	27WF84
103	6	do	21WF55	18WF45	18WF45
104	6	do	24WF68	24WF68	27WF84
105	6	do	do	21WF55	21WF55
106	6	do	18WF45	18WF45	18WF45
107	6	do	do	do	do
108	6	do	do	do	do
117	7	do	do	do	do
118	7	do	do	do	do
119	7	do	do	do	do
120	7	do	27WF84	24WF76	27WF84
121	7	do	18WF45	18WF45	18WF45
122	7	do	27WF84	24WF68	27WF84
123	7	do	21WF55	21WF55	21WF55
124	7	do	18WF45	18WF45	18WF45
125	7	do	do	do	do
126	7	do	do	do	do
135	8	do	do	do	do
136	8	do	do	do	do
137	8	do	do	do	do
138	8	do	27WF84	24WF68	27WF84
139	8	do	18WF45	18WF45	18WF45
140	8	do	24WF76	24WF68	24WF68
141	8	do	21WF55	18WF45	21WF55
142	8	do	18WF45	do	18WF45
143	8	do	do	do	do
144	8	do	do	do	do

Table 5.7. Continued

Member Number	Story Number	Initia Sizes	Final Design Example B1	Final Design Example B2	Final Design Example B3
153	9	16WF36	16WF36	16WF36	16WF36
154	9	do	do	do	do
155	9	do	do	do	do
156	9	do	27WF84	24WF68	27WF84
157	9	do	16WF36	16WF36	16WF36
158	9	do	24WF68	21WF55	24WF68
159	9	do	18WF45	18WF45	18WF45
160	9	do	16WF36	16WF36	16WF36
161	9	do	do	do	do
162	9	do	do	do	do
171	10	do	do	do	do
172	10	do	do	do	do
173	10	do	do	do	do
174	10	do	21WF55	21WF55	24WF68
175	10	do	16WF36	16WF36	16WF36
176	10	do	21WF55	18WF45	21WF55
177	10	do	18WF45	do	18WF45
178	10	do	16WF36	16WF36	16WF36
179	10	do	do	do	do
180	10	do	do	do	do
189	11	do	16WF40	do	do
190	11	do	16WF36	do	do
191	11	do	do	do	do
192	11	do	do	do	do
193	11	do	do	do	do
194	11	do	18WF45	18WF45	16WF40
195	11	do	16WF36	16WF36	16WF36
196	11	do	do	do	do
197	11	do	do	do	do
198	11	do	do	do	do

Table 5.8. Column Sizes - Braced Frame B

Member Number	Story Number	Initial Sizes	Final Design Example BW2	Final Design Example BRW2
1	1	14WF150	14WF150	14WF150
2	1	do	do	do
3	1	do	do	do
4	1	do	do	do
5	1	do	do	do
6	1	do	do	do
7	1	do	do	do
8	1	do	do	do
19	2	do	do	do
20	2	do	do	do
21	2	do	do	do
22	2	do	do	do
23	2	do	do	do
24	2	do	do	do
25	2	do	do	do
26	2	do	do	do
37	3	14WF111	14WF111	14WF111
38	3	do	do	do
39	3	do	do	do
40	3	do	do	do
41	3	do	do	do
42	3	do	do	do
43	3	do	do	do
44	3	do	do	do
55	4	do	do	do
56	4	do	do	do
57	4	do	do	do
58	4	do	do	do
59	4	do	do	do
60	4	do	do	do
61	4	do	do	do
62	4	do	do	do
73	5	14WF74	14WF74	14WF74
74	5	do	do	do
75	5	do	do	do
76	5	do	14WF78	14WF78
77	5	do	14WF84	14WF84
78	5	do	14WF78	14WF78
79	5	do	14WF74	14WF74
80	5	do	14WF78	14WF78

Table 5.8. Continued

Member Number	Story Number	Initial Sizes	Final Design Example BW2	Final Design Example BRW2
91	6	14WF74	14WF74	14WF74
92	6	do	do	do
93	6	do	do	do
94	6	do	14WF78	14WF78
95	6	do	14WF84	14WF84
96	6	do	14WF78	14WF78
97	6	do	14WF74	14WF74
98	6	do	14WF78	14WF78
109	7	12WF40	12WF40	12WF40
110	7	do	14WF43	14WF43
111	7	do	12WF40	12WF40
112	7	do	do	do
113	7	do	14WF43	14WF43
114	7	do	12WF40	12WF40
115	7	do	do	do
116	7	do	do	do
127	8	do	do	do
128	8	do	14WF43	14WF43
129	8	do	12WF40	12WF40
130	8	do	do	do
131	8	do	14WF43	14WF43
132	8	do	12WF40	12WF40
133	8	do	do	do
134	8	do	do	do
145	9	8WF31	8WF31	8WF31
146	9	do	do	do
147	9	do	do	do
148	9	do	do	do
149	9	do	12WF40	14WF43
150	9	do	8WF31	8WF31
151	9	do	do	do
152	9	do	do	do
163	10	do	do	do
164	10	do	do	do
165	10	do	do	do
166	10	do	do	do
167	10	do	do	do
168	10	do	do	do
169	10	do	do	do
170	10	do	do	do

Table 5.8. Continued

Member Number	Story Number	Initial Sizes	Final Design Example BW2	Final Design Example BRW2
181	11	8WF24	8WF24	8WF24
182	11	do	do	do
183	11	do	do	do
184	11	do	do	do
185	11	do	do	do
186	11	do	do	do
187	11	do	do	do
188	11	do	do	do

Table 5.9. Beam Sizes - Braced Frame B

Member Number	Story Number	Initial Sizes	Final Design Example BW2	Final Design Example BRW2
9	1	18WF45	18WF45	18WF45
10	1	do	do	do
11	1	do	do	do
12	1	do	21WF55	21WF55
13	1	do	18WF45	18WF45
14	1	do	21WF55	do
15	1	do	18WF45	do
16	1	do	do	do
17	1	do	do	do
18	1	do	do	do
27	2	do	do	do
28	2	do	do	do
29	2	do	do	do
30	2	do	21WF55	21WF55
31	2	do	18WF45	18WF45
32	2	do	21WF55	21WF55
33	2	do	18WF45	18WF45
34	2	do	do	do
35	2	do	do	do
36	2	do	do	do
45	3	do	do	do
46	3	do	do	do
47	3	do	do	do
48	3	do	21WF55	21WF55
49	3	do	18WF45	18WF45
50	3	do	21WF55	21WF55
51	3	do	18WF45	18WF45
52	3	do	do	do
53	3	do	do	do
54	3	do	do	do
63	4	do	do	do
64	4	do	do	do
65	4	do	do	do
66	4	do	21WF55	21WF55
67	4	do	18WF45	18WF45
68	4	do	21WF55	21WF55
69	4	do	18WF45	18WF45
70	4	do	do	do
71	4	do	do	do
72	4	do	do	do

Table 5.9. Continued

Member Number	Story Number	Initial Sizes	Final Design Example BW2	Final Design Example BRW2
81	5	16WF36	16WF36	16WF36
82	5	do	do	do
83	5	do	do	do
84	5	do	18WF45	18WF45
85	5	do	16WF36	16WF36
86	5	do	18WF45	18WF45
87	5	do	16WF36	16WF36
88	5	do	do	do
89	5	do	do	do
90	5	do	do	do
99	6	do	do	do
100	6	do	do	do
101	6	do	do	do
102	6	do	18WF45	18WF45
103	6	do	16WF36	16WF36
104	6	do	do	do
105	6	do	do	do
106	6	do	do	do
107	6	do	do	do
108	6	do	do	do
117	7	do	do	do
118	7	do	do	do
119	7	do	do	do
120	7	do	do	do
121	7	do	do	do
122	7	do	do	do
123	7	do	do	do
124	7	do	do	do
125	7	do	do	do
126	7	do	do	do
135	8	do	do	do
136	8	do	do	do
137	8	do	do	do
138	8	do	do	do
139	8	do	do	do
140	8	do	do	do
141	8	do	do	do
142	8	do	do	do
143	8	do	do	do
144	8	do	do	do

Table 5.9. Continued

Member Number	Story Number	Initial Sizes	Final Design Example BW2	Final Design Example BRW2
153	9	16B26	14WF30	14WF30
154	9	do	16B26	16B26
155	9	do	14WF30	14WF30
156	9	do	16WF36	16WF36
157	9	do	16B26	16B26
158	9	do	16WF36	16WF36
159	9	do	16B26	16B26
160	9	do	do	do
161	9	do	do	do
162	9	do	do	do
171	10	do	14WF30	14WF30
172	10	do	16B26	16B26
173	10	do	14WF30	14WF30
174	10	do	16WF36	16WF36
175	10	do	16B26	16B26
176	10	do	14WF34	14WF34
177	10	do	16B26	16B26
178	10	do	do	do
179	10	do	do	do
180	10	do	do	do
189	11	do	14WF30	14WF30
190	11	do	16B26	16B26
191	11	do	14WF30	14WF30
192	11	do	do	do
193	11	do	16B26	16B26
194	11	do	14WF30	14WF30
195	11	do	16B26	16B26
196	11	do	do	do
197	11	do	do	do
198	11	do	do	do

Table 5.10. Brace Sizes - Braced Frame B

Member Number	Story Number	Initial Sizes	Final Design Example BW2	Final Design Example BRW2
199	1	4UAN11.6	4UAN11.6	4UAN11.6
200	2	do	do	do
201	3	do	do	do
202	4	do	do	do
203	5	3UAN9.0	3UAN9.0	3UAN9.0
204	6	do	do	do
205	7	do	do	do
206	8	do	do	do
207	9	3UAN6.1	3UAN6.1	3UAN6.1
208	10	do	do	do
209	11	do	do	do
210	1	4UAN11.6	4UAN21.2	6UAN23.4
211	2	do	6UAN24.6	5UAN27.2
212	3	do	4UAN18.2	4UAN19.6
213	4	do	4UAN17.0	4UAN18.2
214	5	3UAN9.0	do	4UAN19.6
215	6	do	do	4UAN18.2
216	7	do	4UAN18.2	4UAN19.6
217	8	do	4UAN15.4	4UAN17.0
218	9	3UAN6.1	4UAN14.4	4UAN15.4
219	10	do	4UAN11.6	4UAN11.6
220	11	do	3UAN9.0	3UAN9.0
221	1	4UAN11.6	4UAN11.6	4UAN11.6
222	2	do	do	do
223	3	do	do	do
224	4	do	do	do
225	5	3UAN9.0	3UAN9.0	3UAN9.0
226	6	do	do	do
227	7	do	do	do
228	8	do	do	do
229	9	3UAN6.1	3UAN6.1	3UAN6.1
230	10	do	do	do
231	11	do	do	do
232	1	4UAN11.6	4UAN11.6	4UAN11.6
233	2	do	do	do
234	3	do	do	do
235	4	do	do	do
236	5	3UAN9.0	3UAN9.0	3UAN9.0
237	6	do	do	do
238	7	do	do	do
239	8	do	do	do
240	9	3UAN6.1	3UAN6.1	3UAN6.1
241	10	do	do	do

Table 5.10. Continued

Member Number	Story Number	Initial Sizes	Final Design Example BW2	Final Design Example BRW2
242	11	3UAN6.1	3UAN6.1	3UAN6.1
243	1	4UAN11.6	4UAN19.6	4UAN19.6
244	2	do	6UAN23.4	4UAN22.2
245	3	do	4UAN17.0	4UAN17.0
246	4	do	do	4UAN15.4
247	5	3UAN9.0	do	4UAN17.0
248	6	do	4UAN15.4	4UAN15.4
249	7	do	4UAN17.0	4UAN17.0
250	8	do	4UAN14.4	4UAN14.4
251	9	3UAN6.1	3UAN13.2	3UAN13.2
252	10	do	4UAN11.6	4UAN11.6
253	11	do	3UAN9.0	3UAN9.0
254	1	4UAN11.6	4UAN11.6	4UAN11.6
255	2	do	do	do
256	3	do	do	do
257	4	do	do	do
258	5	3UAN9.0	3UAN9.0	3UAN9.0
259	6	do	do	do
260	7	do	do	do
261	8	do	do	do
262	9	3UAN6.1	3UAN6.1	3UAN6.1
263	10	do	do	do
264	11	do	do	do

Table 5.11 Summary of Frame A Results

Example (Group)	Displacement Constraint #	Displacement Constraint (d) (inches)	Initial Displ. (Abs. Value) (inches)	Final Displ. (Abs. Value) (f) (inches)	% of (d) below (f)	Initial Error, % (EP)	Column Changes	Beam Changes	Brace Changes	Total Changes	No. of Stiffness Analyses	CPU Time (seconds)	Initial Weight (tons)	Final Weight (tons)	Initial Cost ($)	Final Cost ($)
A1 (1)	1	6.91	8.65	6.00	13.2	25	7	183	0	190	3	136.7	155.86	168.27	62,345	67,309
	2	4.90	6.51	4.51	8.0											
	3	6.91	8.64	6.00	13.2											
A2 (1)	1	6.91	8.65	6.50	5.9	10	7	151	0	158	3	134.9	155.86	164.76	62,345	65,905
	2	4.90	6.51	4.73	3.5											
	3	6.91	8.64	6.49	6.1											
A3 (1)	1	6.91	8.65	6.18	10.6	0	7	171	0	178	4	172.7	155.86	166.83	62,345	66,731
	2	4.90	6.51	4.52	7.8											
	3	6.91	8.64	6.17	10.7											

Table 5.11 Continued

Example (Group)	Displacement Constraint #	Displacement Constraint (d) (inches)	Initial Displ. (Abs. Value) (inches)	Final Displ. (Abs. Value) (f) (inches)	% of (d) below (f)	Initial Error, % (EP)	Column Changes	Beam Changes	Brace Changes	Total Changes	No. of Stiffness Analyses	CPU Time (seconds)	Initial Weight (tons)	Final Weight (tons)	Initial Cost ($)	Final Cost ($)
AWL1 (2)	1	6.91	7.04	6.13	11.3	25	4	89	8	101	2	81.7	145.29	147.63	58,116	59,053
	2	4.90	5.35	4.68	4.5											
AWL2 (2)	1	6.91	7.04	6.32	8.5	10	2	50	10	62	2	81.0	145.29	146.91	58,116	58,766
	2	4.90	5.35	4.79	2.2											
AWL3 (2)	1	6.91	7.04	6.46	6.5	0	2	36	7	45	3	113.0	145.29	146.69	58,116	58,678
	2	4.90	5.35	4.86	0.8											
AWR1 (3)	1	6.91	6.97	5.93	14.2	25	4	83	13	100	2	77.0	145.29	148.37	58,116	59,349
	2	4.90	5.29	4.51	8.0											
AWR2 (3)	1	6.91	6.97	6.30	8.8	10	2	38	10	50	2	72.6	145.29	146.63	58,116	58,653
	2	4.90	5.29	4.79	2.2											
AWR3 (3)	1	6.91	6.97	6.42	7.1	0	2	32	5	39	3	103.2	145.29	146.37	58,116	58,549
	2	4.90	5.29	4.85	1.0											

Table 5.12 Summary of Frame B Results

Example (Group)	Displacement Constraint #	Displacement Constraint (d) (inches)	Initial Displ.(Abs. Value) (inches)	Final Displ.(Abs. Value) (f) (inches)	% of (d) below (f)	Initial Error, % (EP)	Column Changes	Beam Changes	Brace Changes	Total Changes	No. of Stiffness Analyses	CPU Time (seconds)	Initial Weight (tons)	Final Weight (tons)	Initial Cost ($)	Final Cost ($)
B1 (4)	1	3.60	5.37	3.15	12.5	25	146	191	0	337	3	176.3	104.76	123.83	41,906	49,533
	2	3.60	5.80	3.21	10.8											
B2 (4)	1	3.60	5.37	3.59	0.3	10	109	138	0	247	3	171.1	104.76	118.02	41,906	47,208
	2	3.60	5.80	3.56	1.1											
B3 (4)	1	3.60	5.37	3.14	12.8	0	148	183	0	331	4	223.2	104.76	123.41	41,906	49,366
	2	3.60	5.80	3.17	11.9											
BW1 (5)	1	3.60	4.69	3.15	12.5	25	30	57	163	250	2	168.9	96.46	103.27	38,586	41,307
	2	3.60	4.79	3.18	11.7											
BW2 (5)	1	3.60	4.69	3.50	2.8	10	19	45	99	163	2	162.4	96.46	101.06	38,586	40,424
	2	3.60	4.79	3.47	3.6											
BW3 (5)	1	3.60	4.69	3.47	3.6	0	12	25	126	163	3	241.5	96.46	100.55	38,586	40,220
	2	3.60	4.79	3.45	4.2											

Table 5.12 Continued

Example (Group)	Displacement Constraint #	Displacement Constraint (d) (inches)	Initial Displ.(Abs. Value) (inches)	Final Displ.(Abs. Value) (f) (inches)	% of (d) below (f)	Initial Error, % (EP)	Column Changes	Beam Changes	Brace Changes	Total Changes	No. of Stiffness Analyses	CPU Time (seconds)	Initial Weight (tons)	Final Weight (tons)	Initial Cost ($)	Final Cost ($)
BRW1 (6)	1	3.60	4.62	3.08	14.4	25	34	59	157	250	2	132.7	96.46	103.32	38,586	41,326
	2	3.60	5.01	3.30	8.3											
BRW2 (6)	1	3.60	4.62	3.47	3.6	10	20	43	109	172	2	130.1	96.46	101.18	38,586	40,473
	2	3.60	5.01	3.48	3.3											
BRW3 (6)	1	3.60	4.62	3.44	4.4	0	18	31	129	178	3	189.6	96.46	101.05	38,586	40,419
	2	3.60	5.01	3.42	5.0											

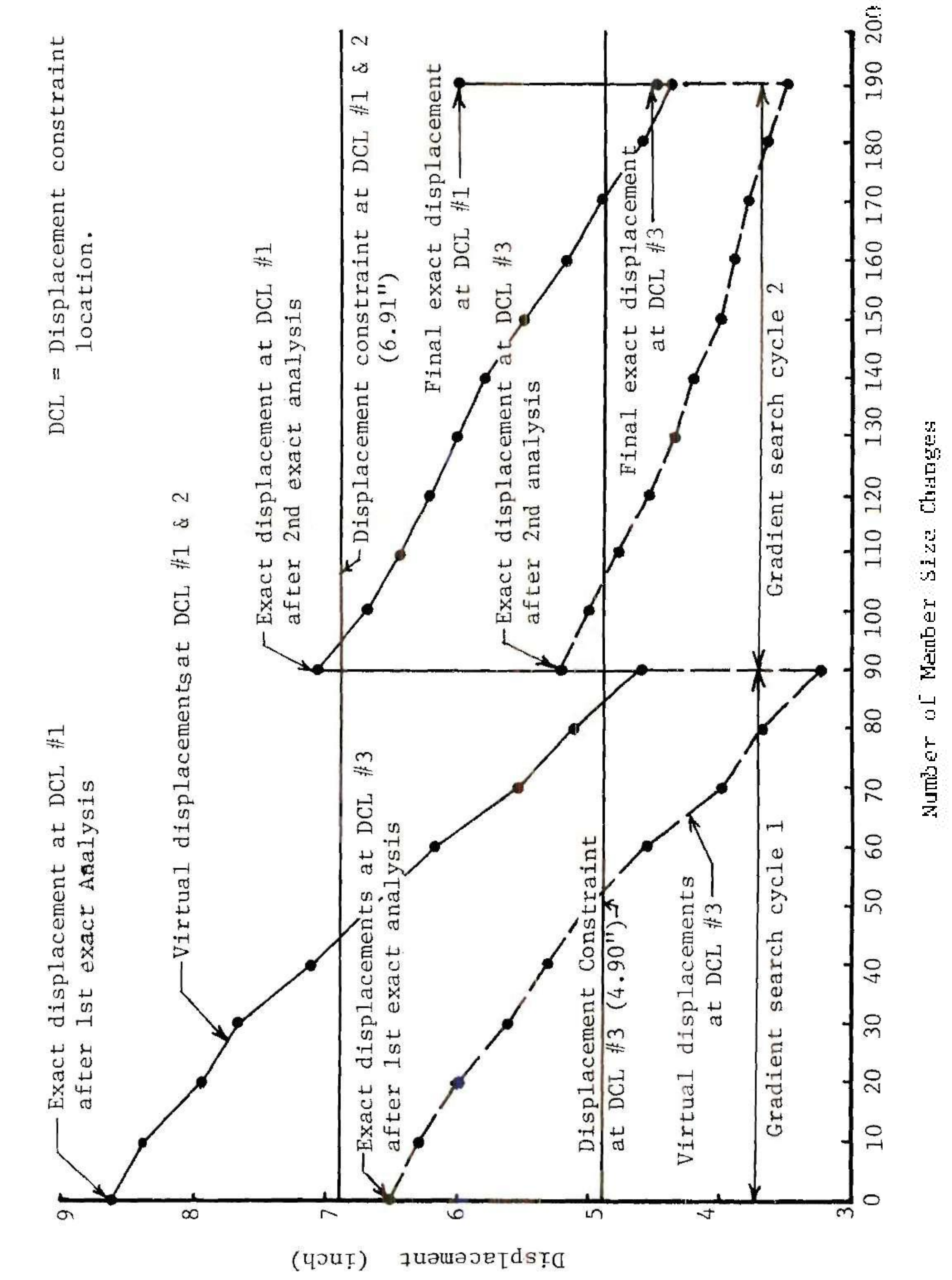


Figure 5.9 Displacements vs. Member Size Changes for Example A1 (EP = 25%)

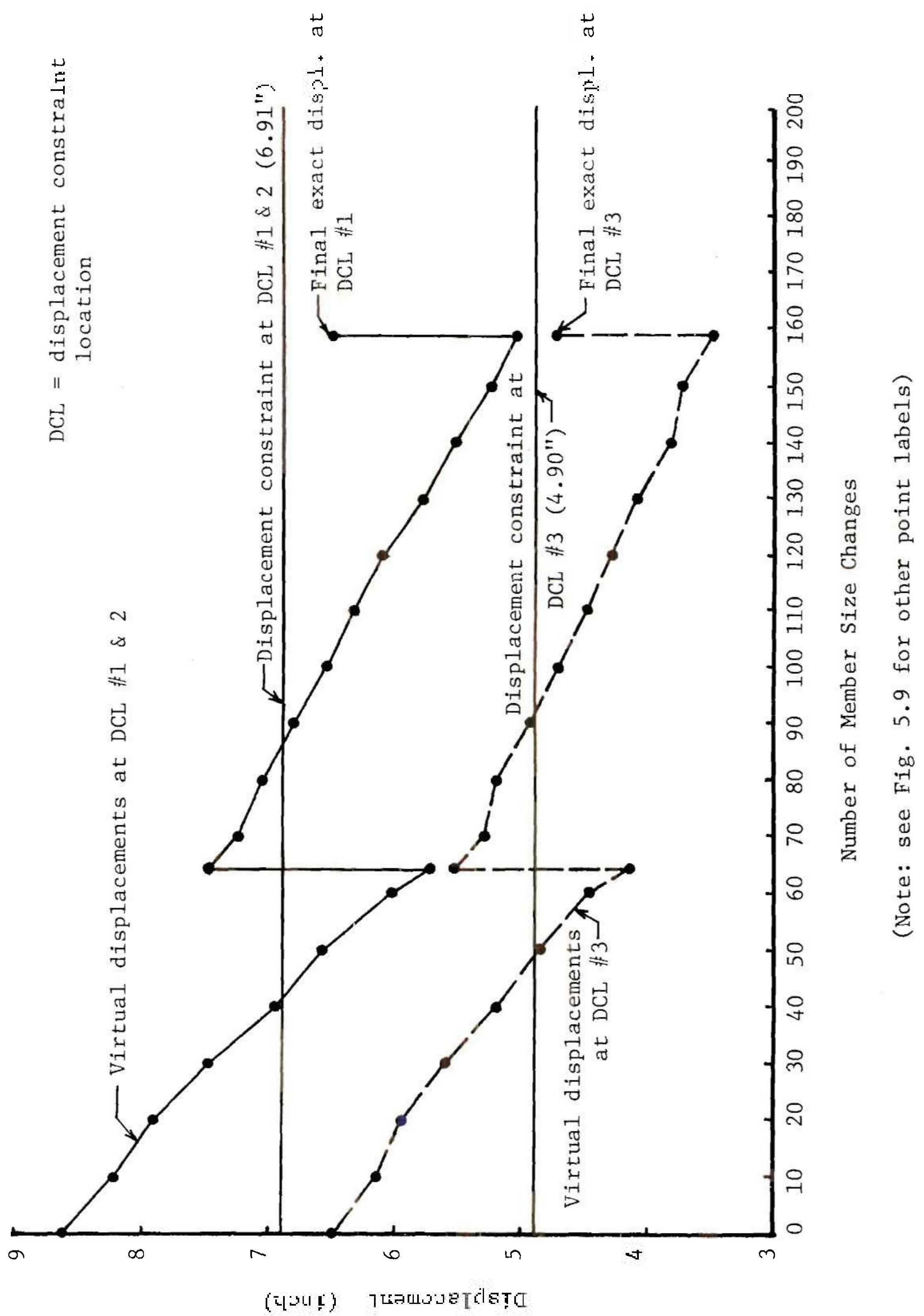


Figure 5.10 Displacements vs. Member Size Changes for Example A2 (EP = 10%)

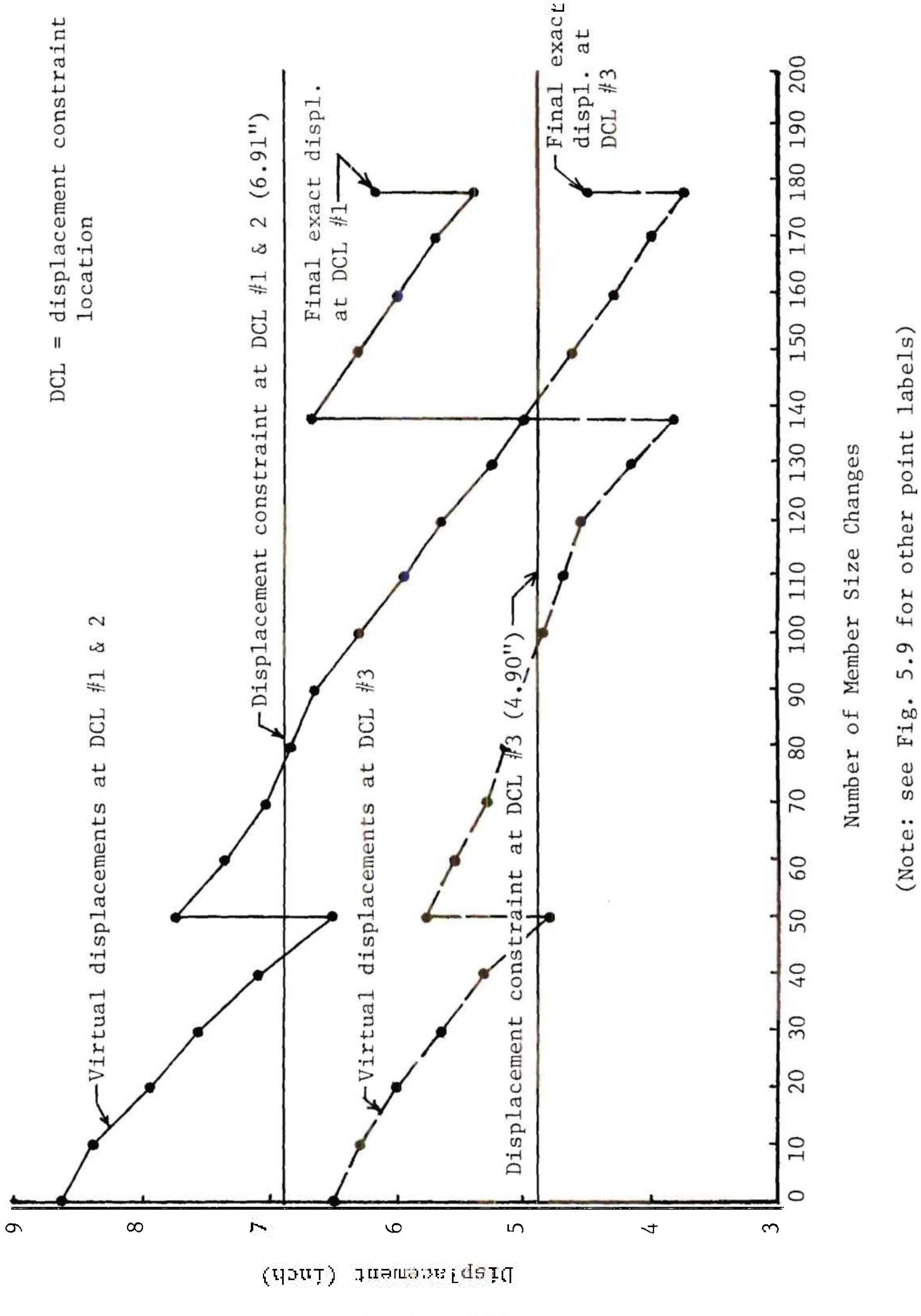


Figure 5.11 Displacements vs. Member Size Changes for Example A3 (EP = 0%)

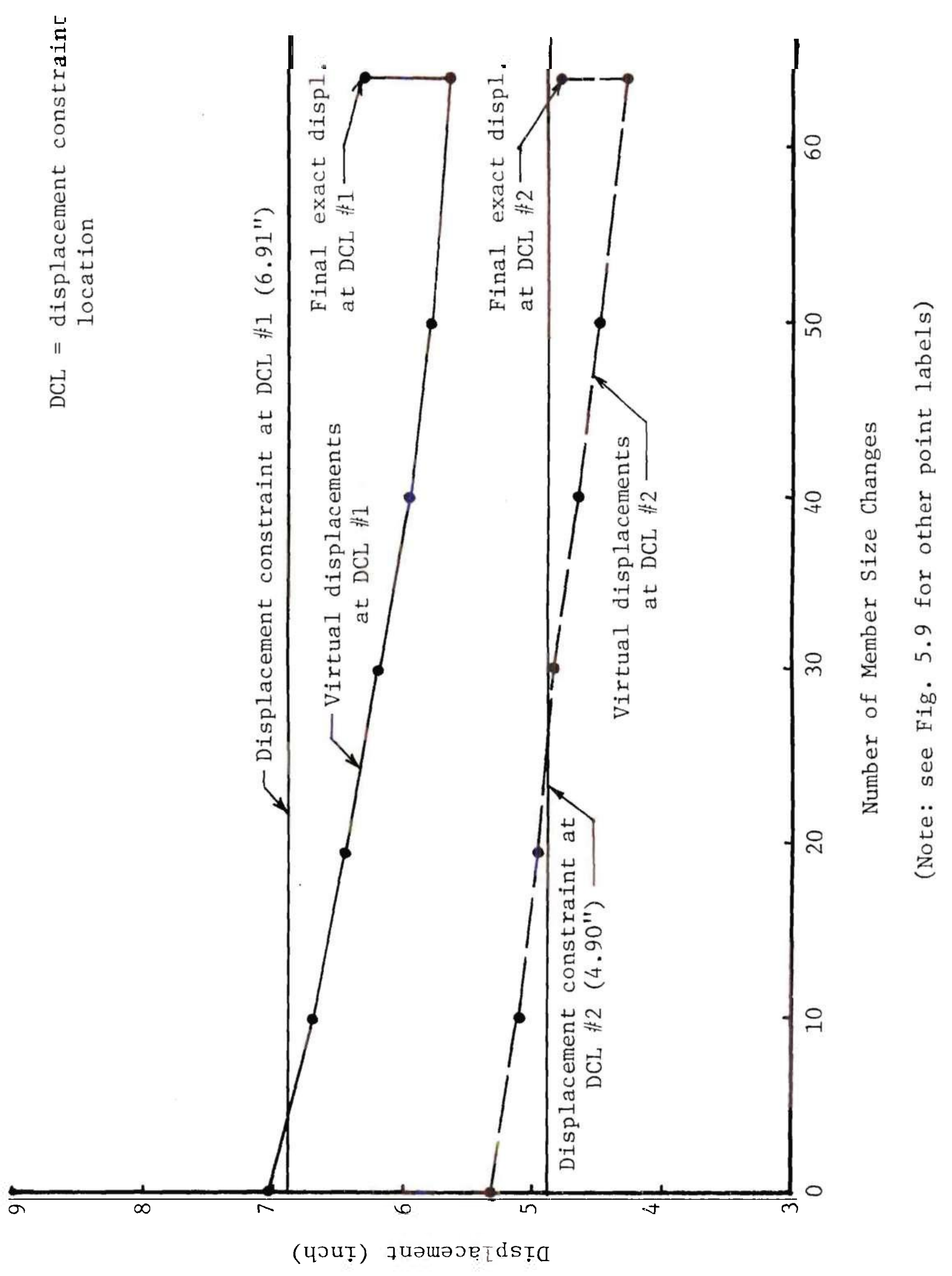


Figure 5.12 Displacements vs. Member Size Changes for Example AWL2 (EP = 10%)

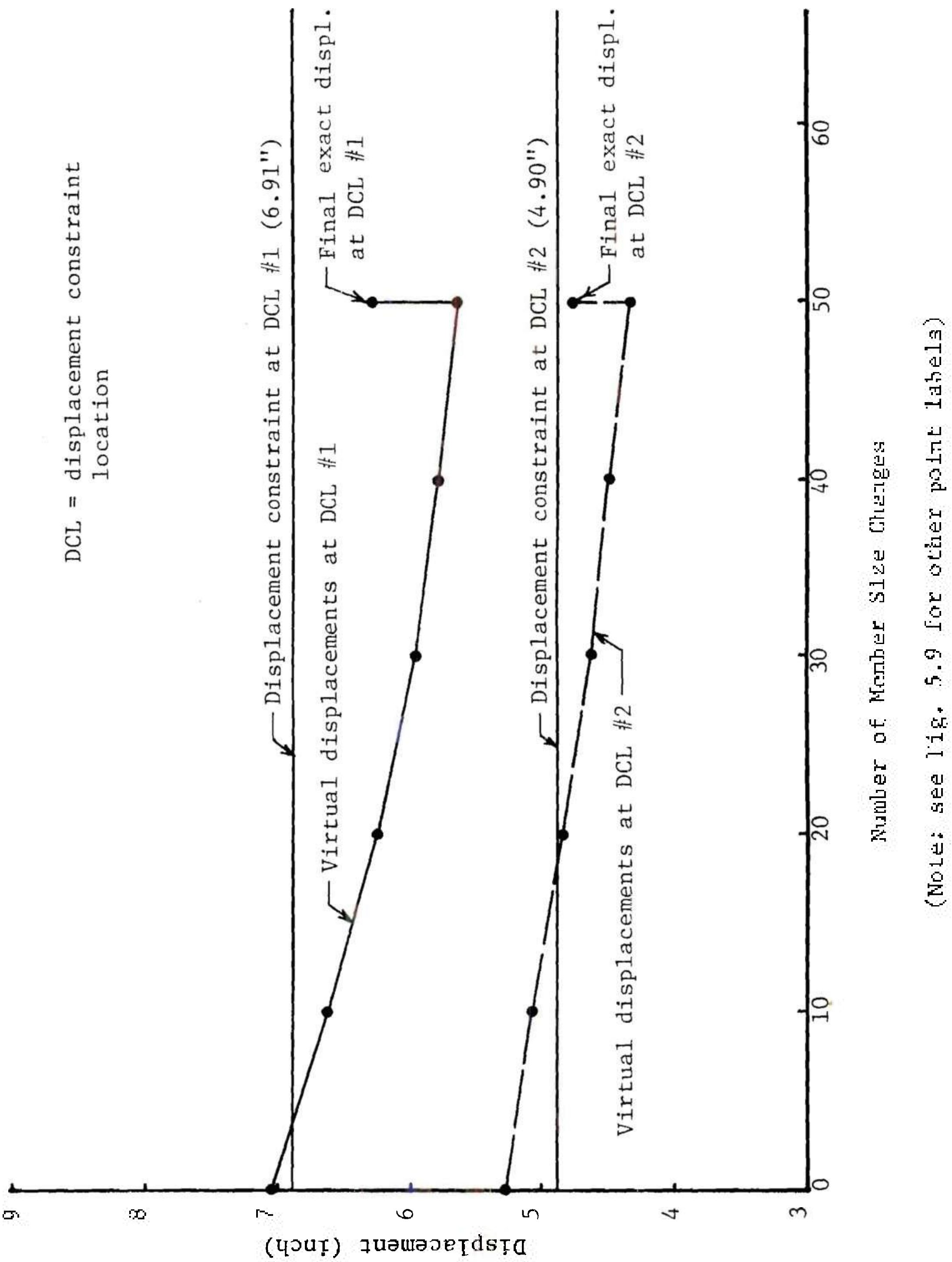


Figure 5.13 Displacements vs. Member Size Changes for Example AWR2 (EP = 10%)

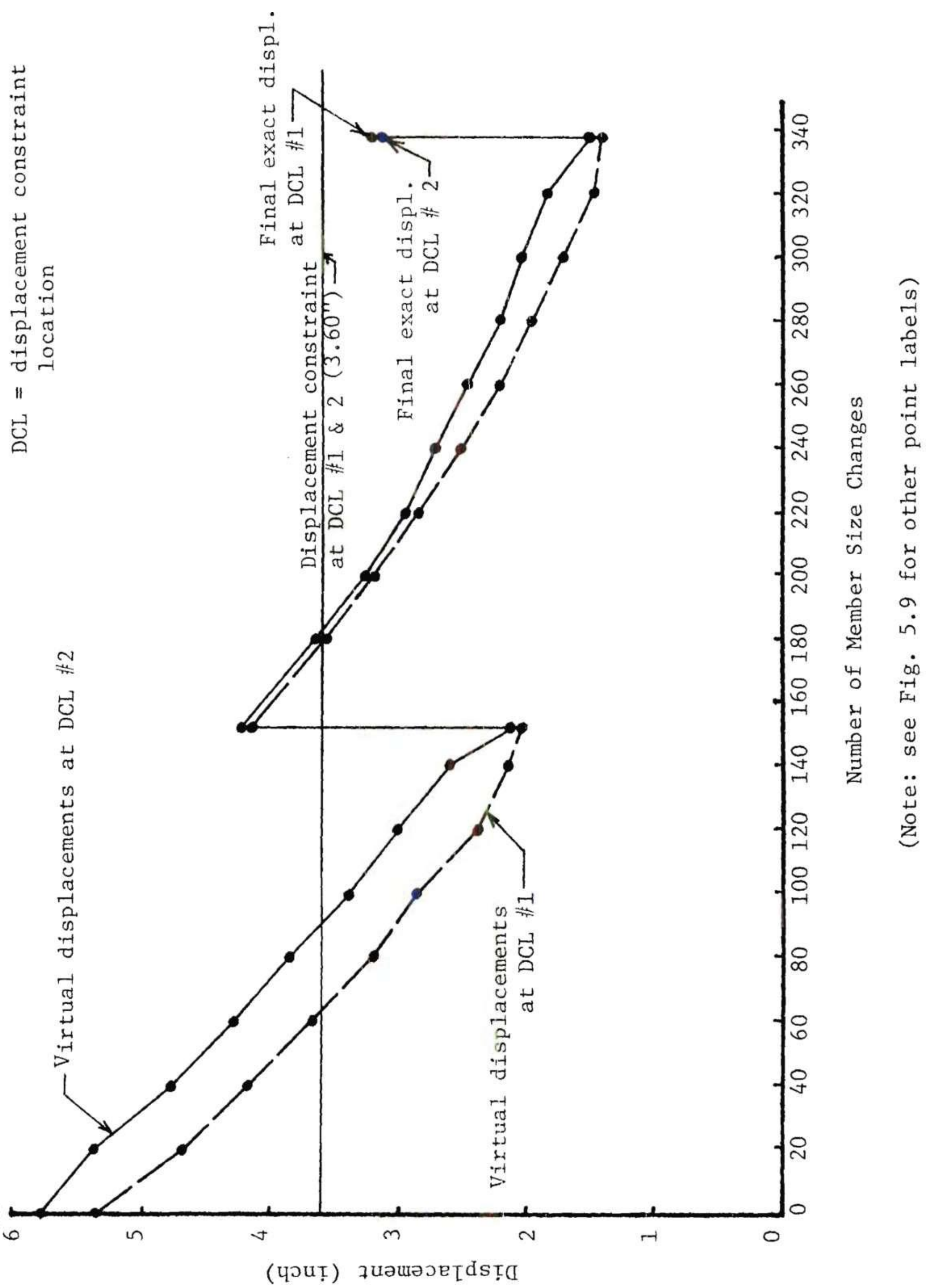


Figure 5.14 Displacements vs. Member Size Changes for Example B1 (EP = 25%)

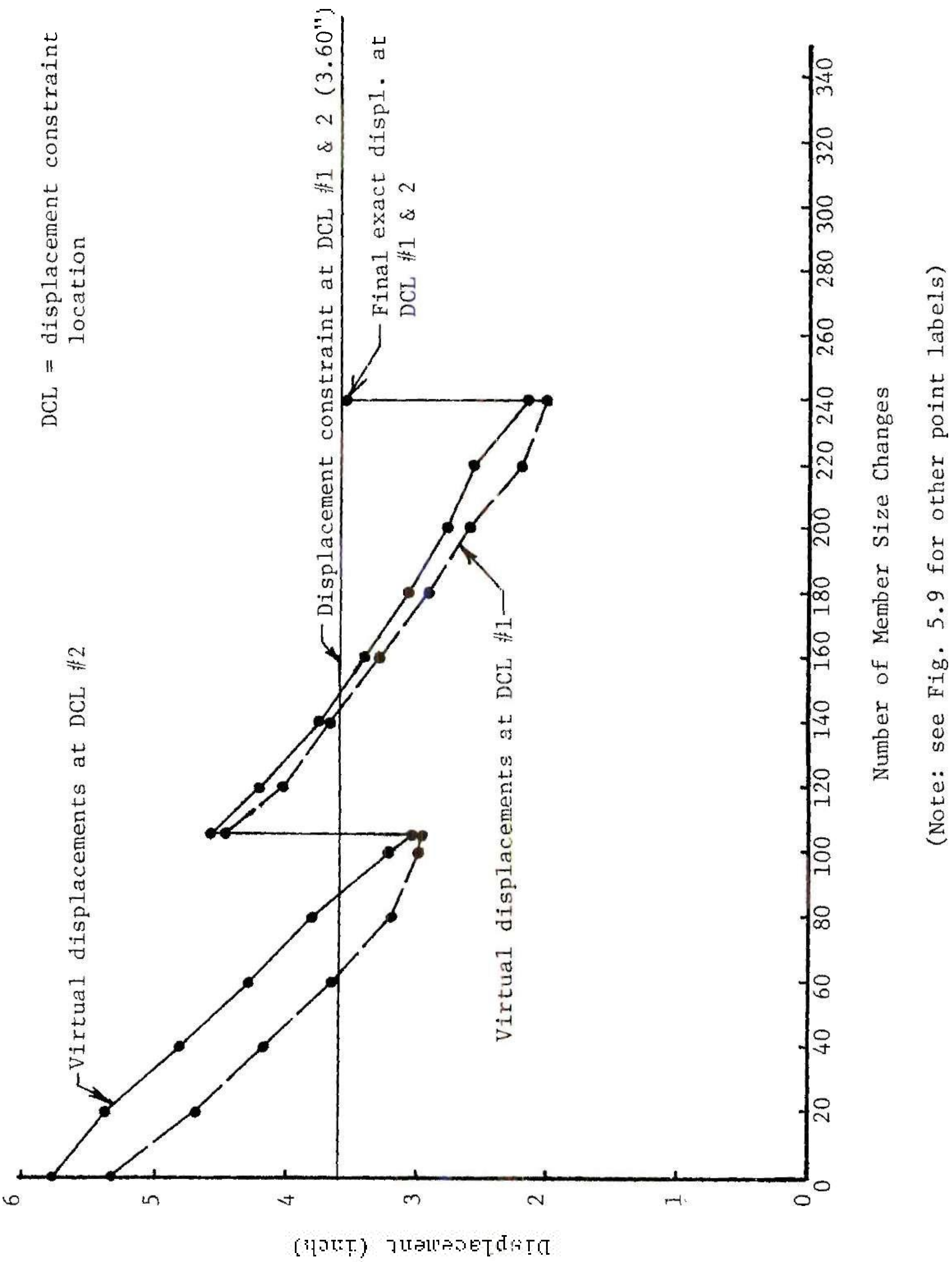


Figure 5.15 Displacements vs. Member Size Changes for Example B2 (EP = 10%)

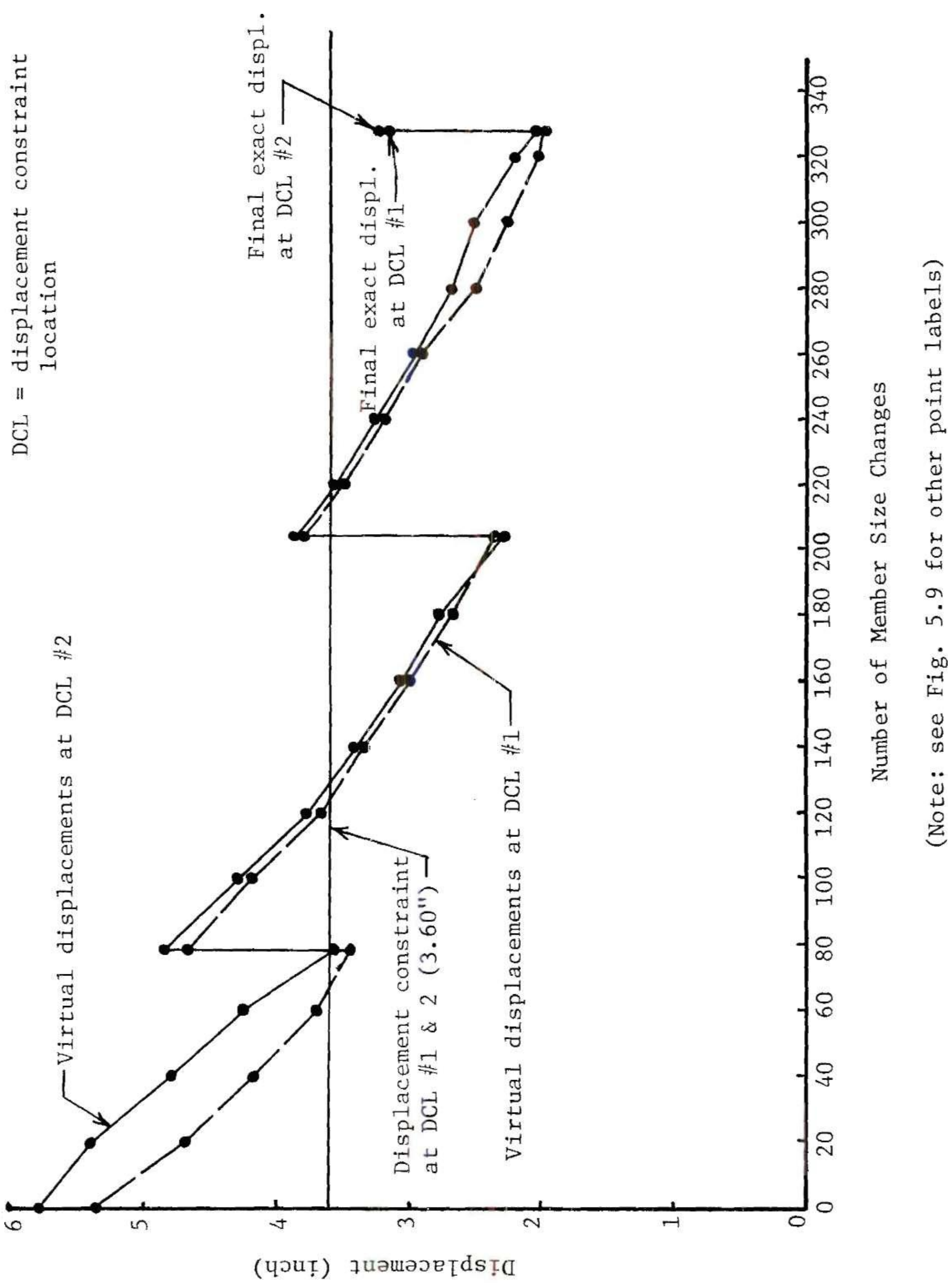


Figure 5.16 Displacements vs. Member Size Changes for Example B3 (EP = 0%)

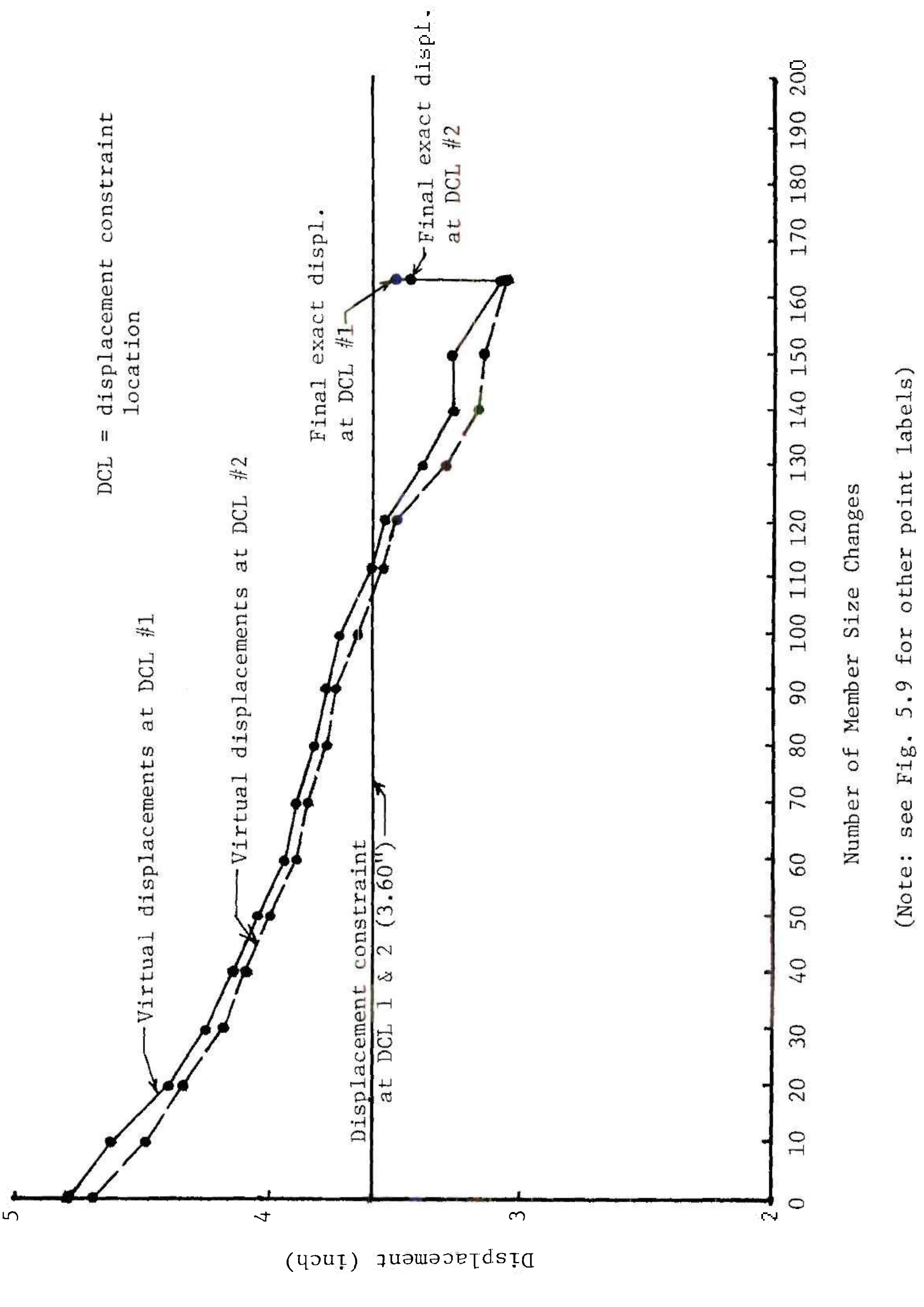


Figure 5.17 Displacements vs. Member Size Changes for Example BW2 (EP = 10%)

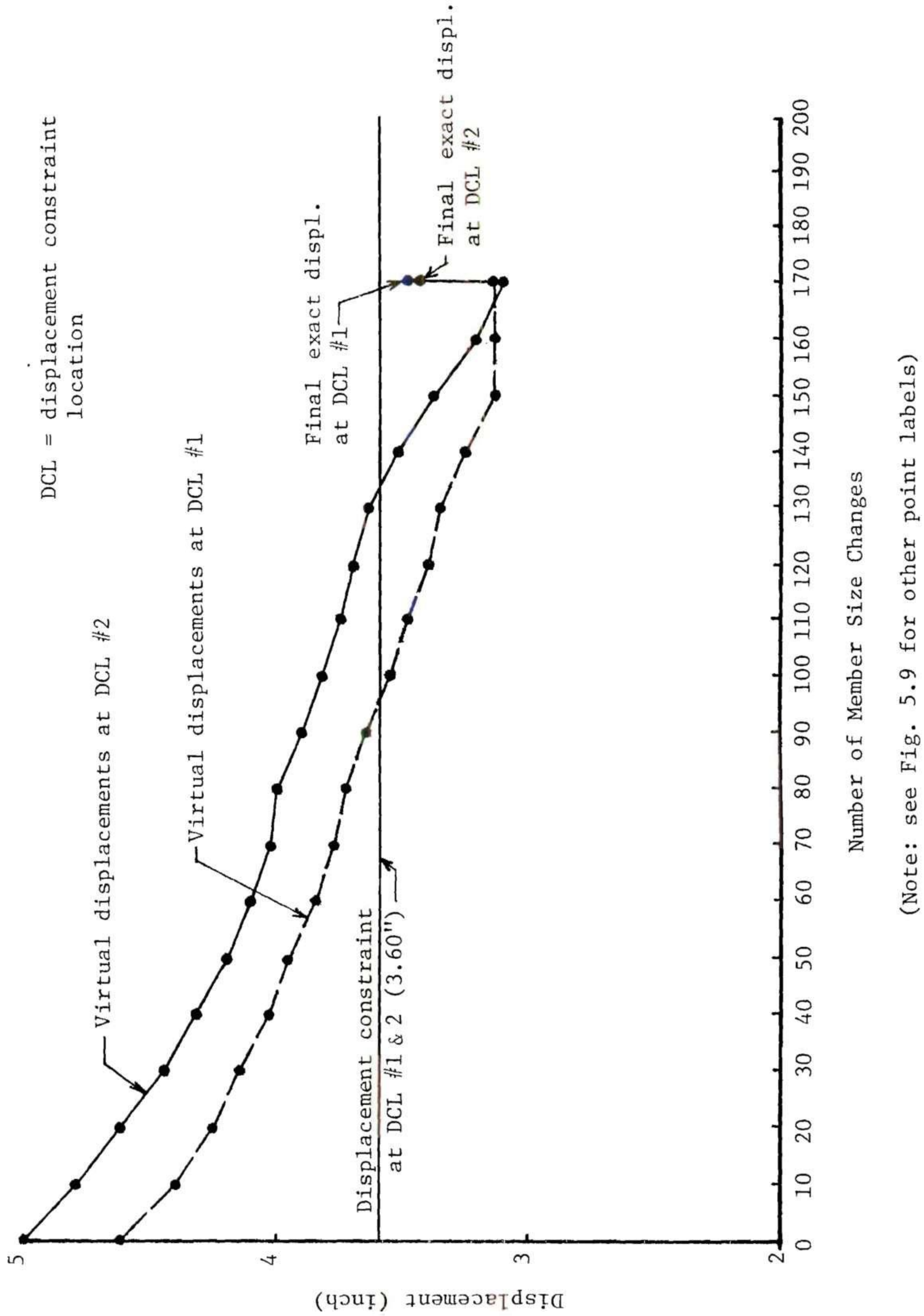


Figure 5.18 Displacements vs. Member Size Changes
for Example BRW2 (EP = 10%)

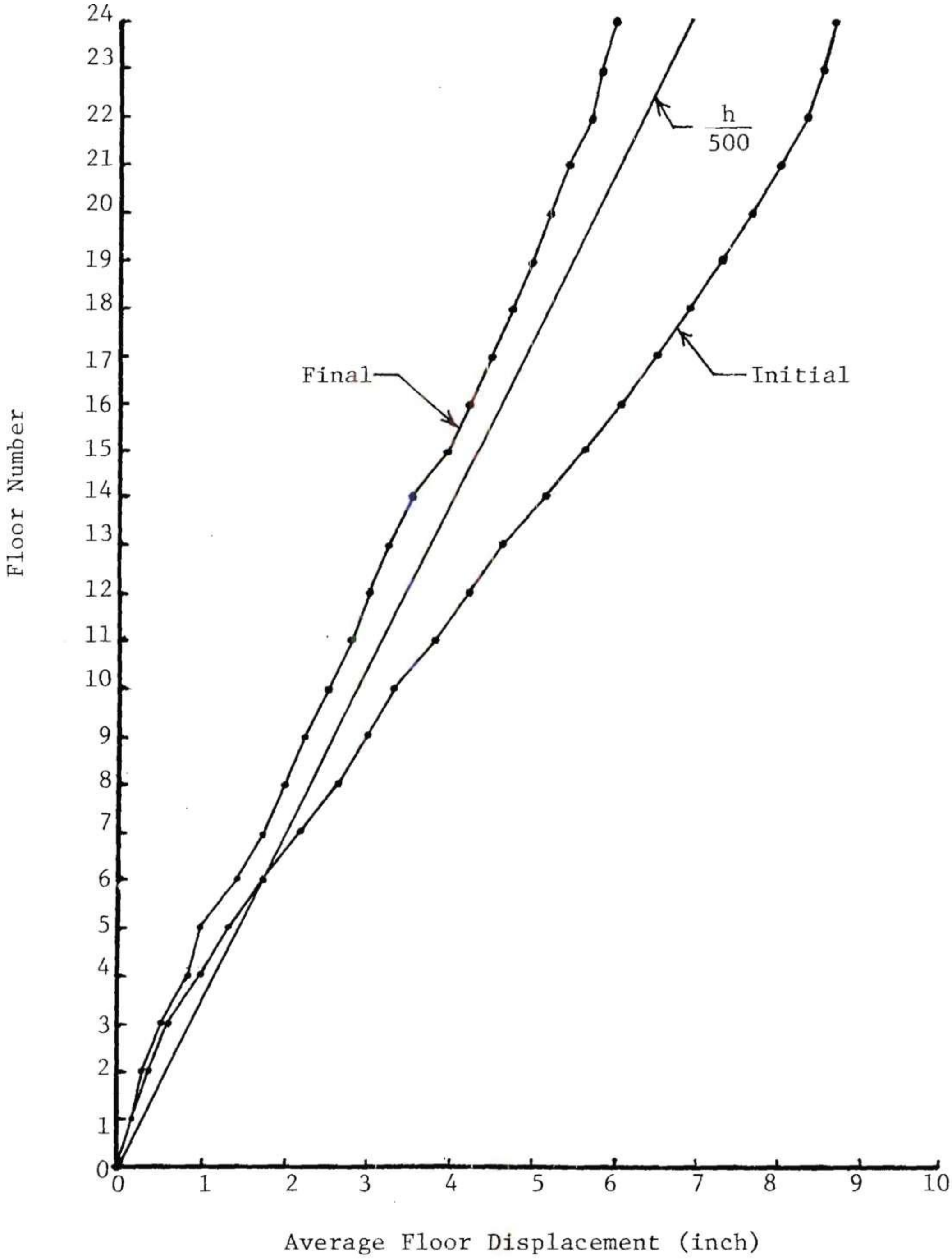


Figure 5.19 Average Floor Displacement - Example A1 (EP = 25%)

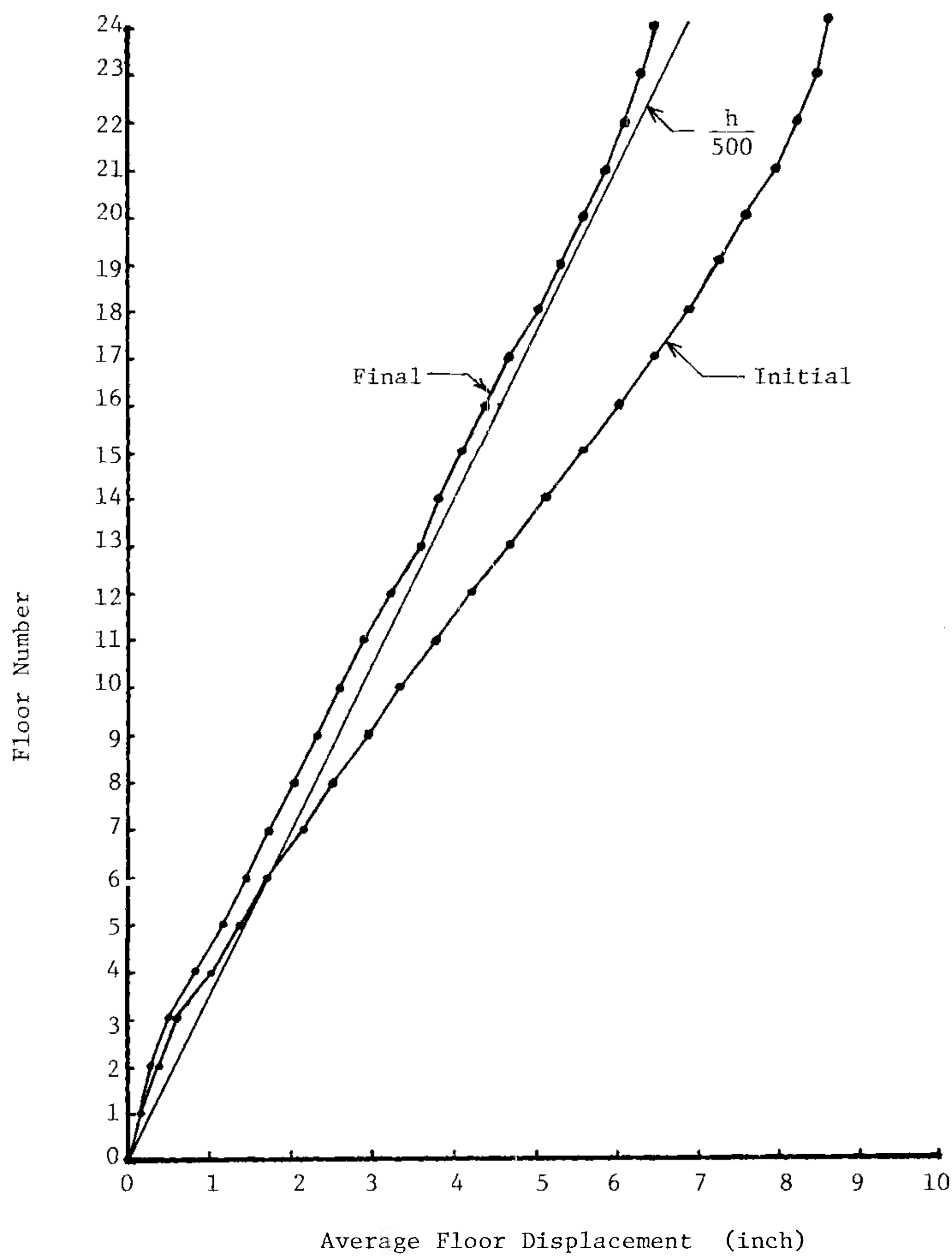


Figure 5.20 Average Floor Displacement - Example A2 (EP = 10%)

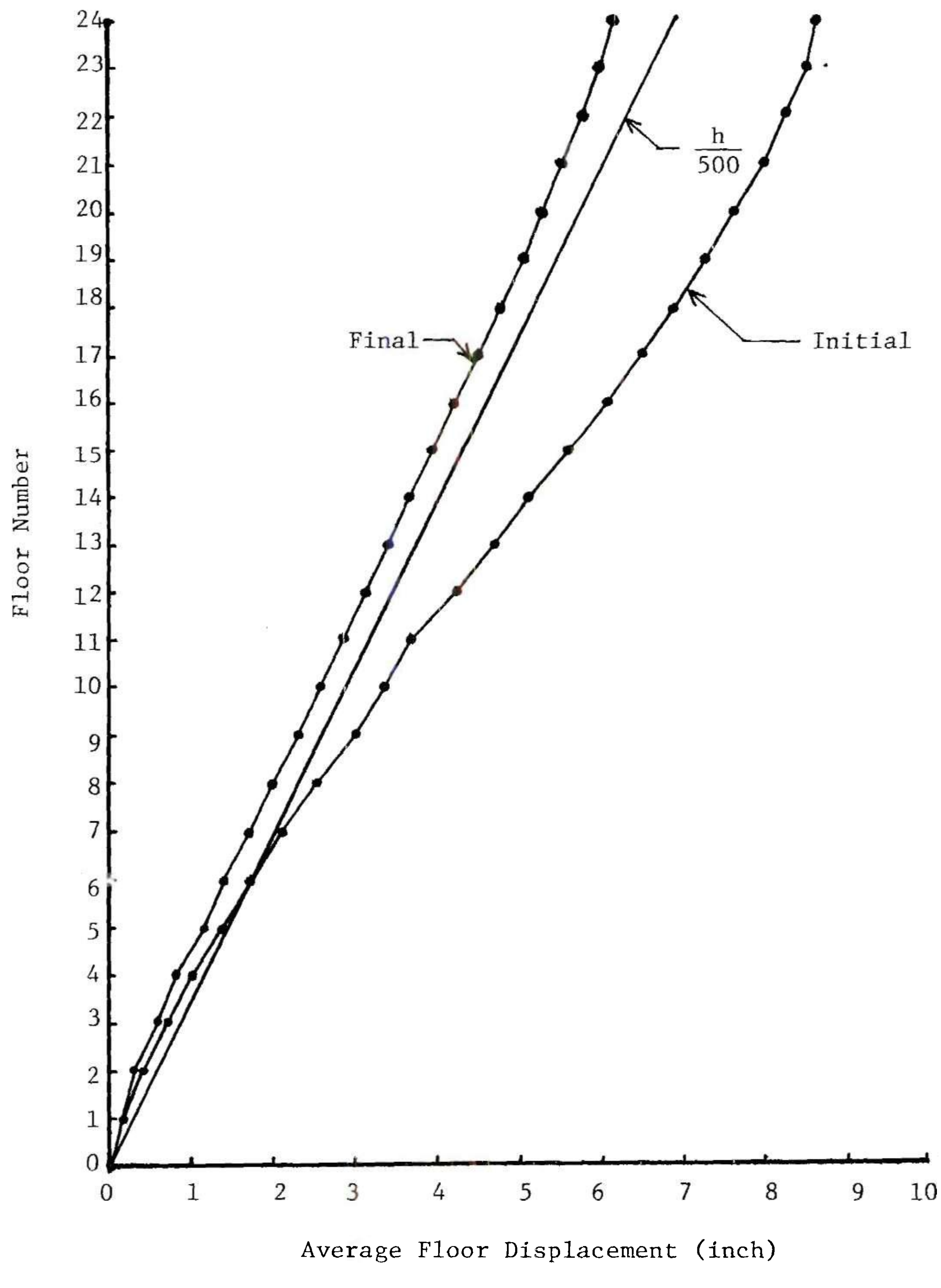


Figure 5.21 Average Floor Displacement - Example A3 (EP = 0%)

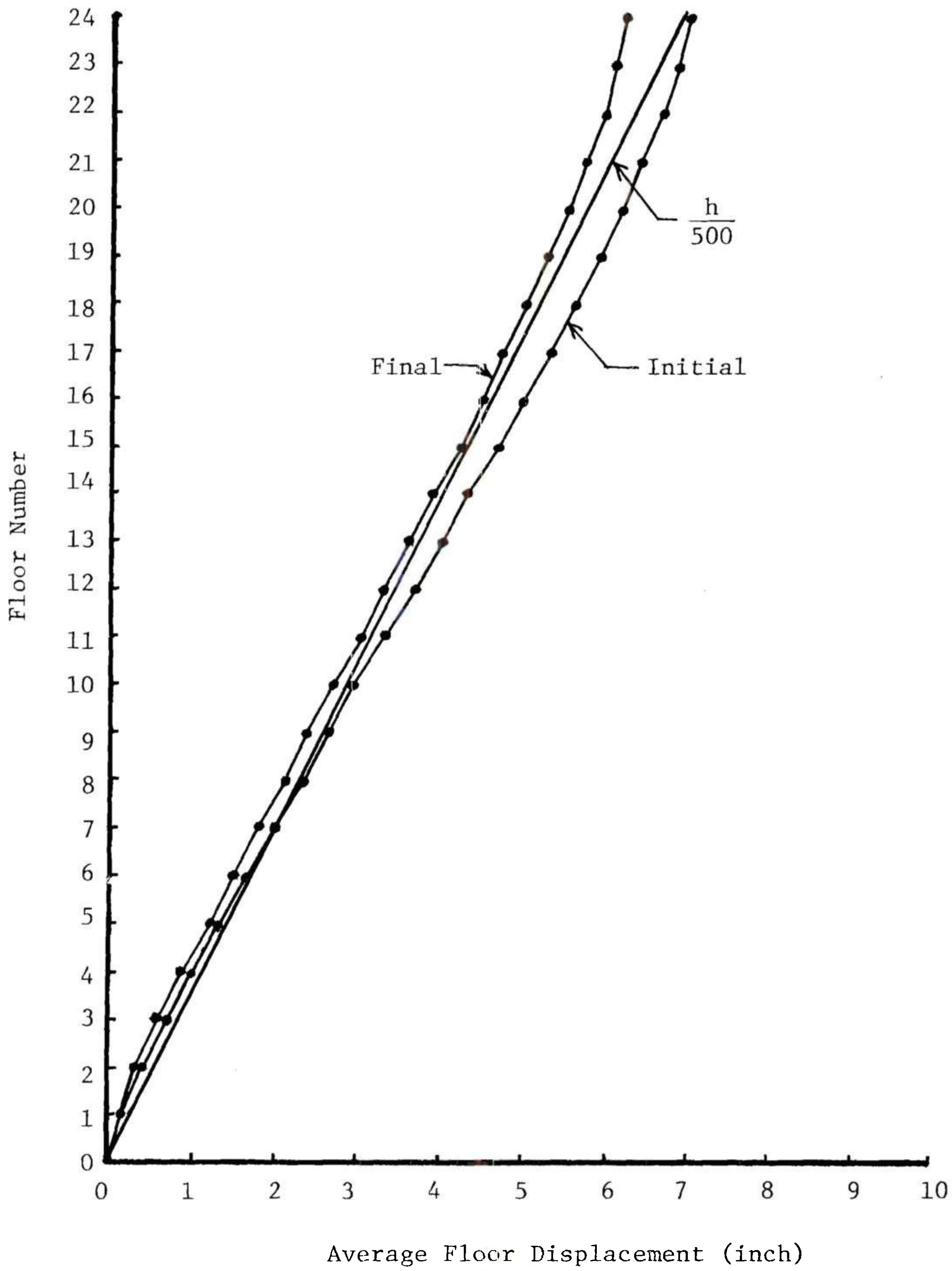


Figure 5.22 Average Floor Displacement - Example AWL2 (EP = 10%)

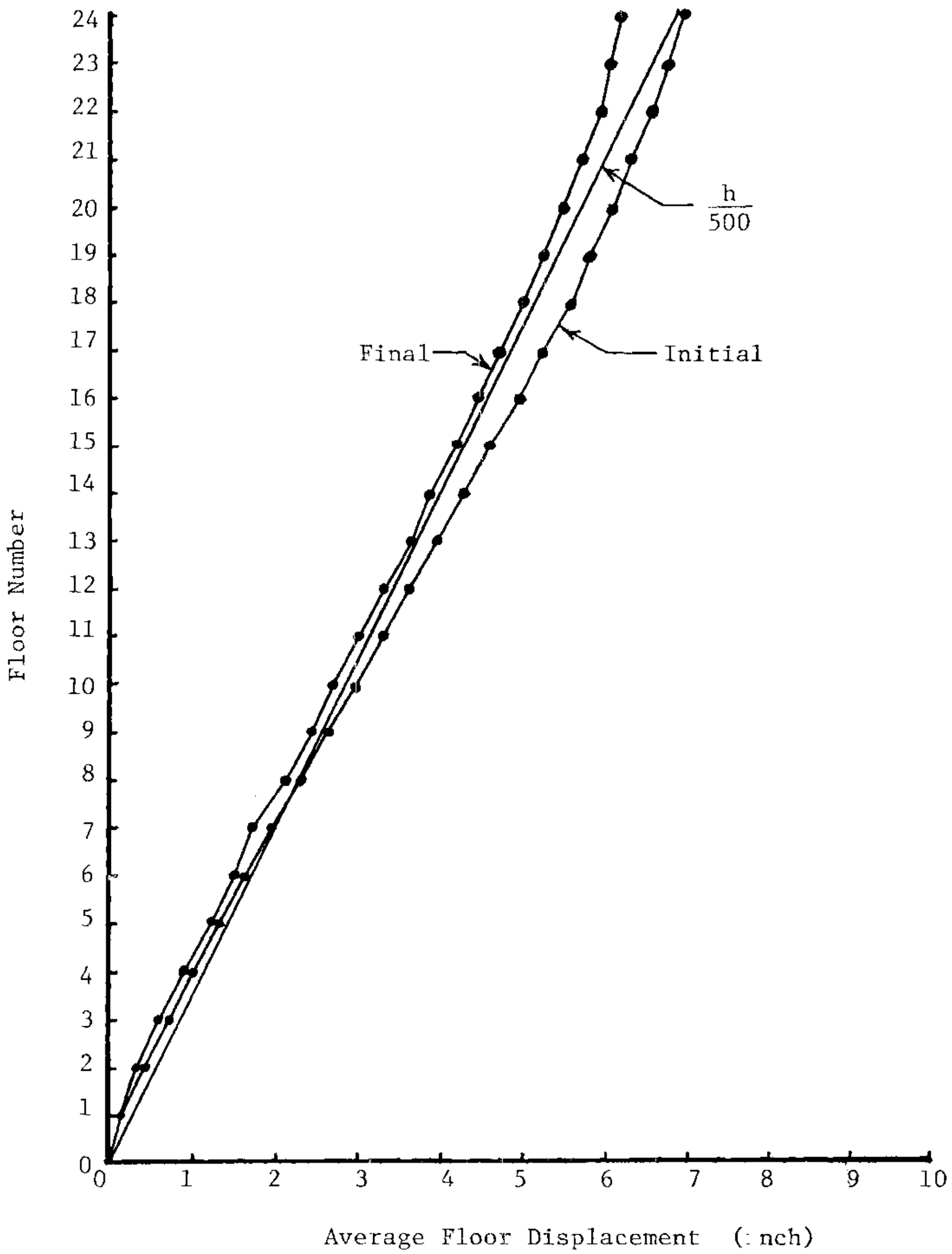


Figure 5.23 Average Floor Displacement - Example AWR2 (EP = 10%)

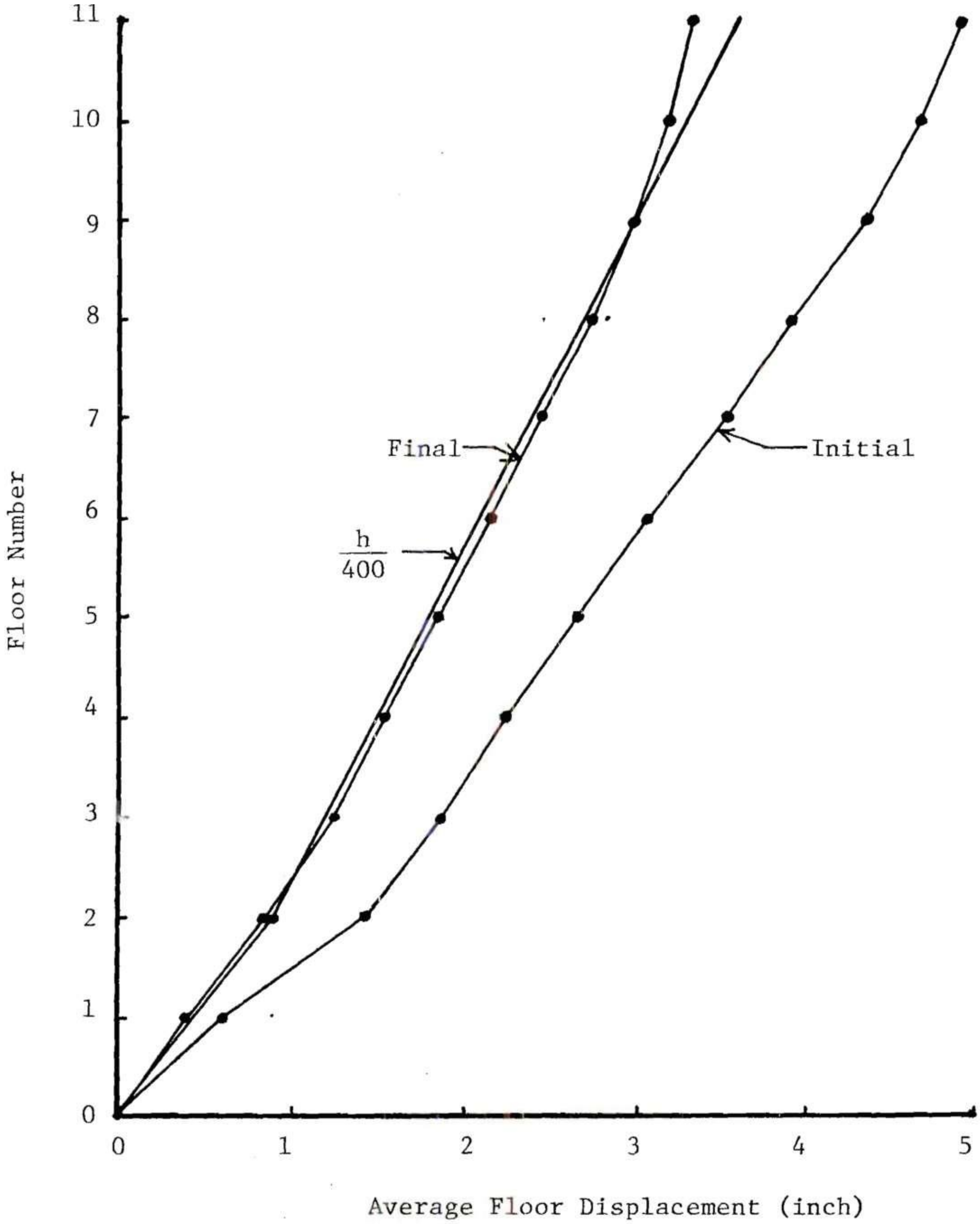


Figure 5.24 Average Floor Displacement, X-Direction - Example B1 (EP = 25%)

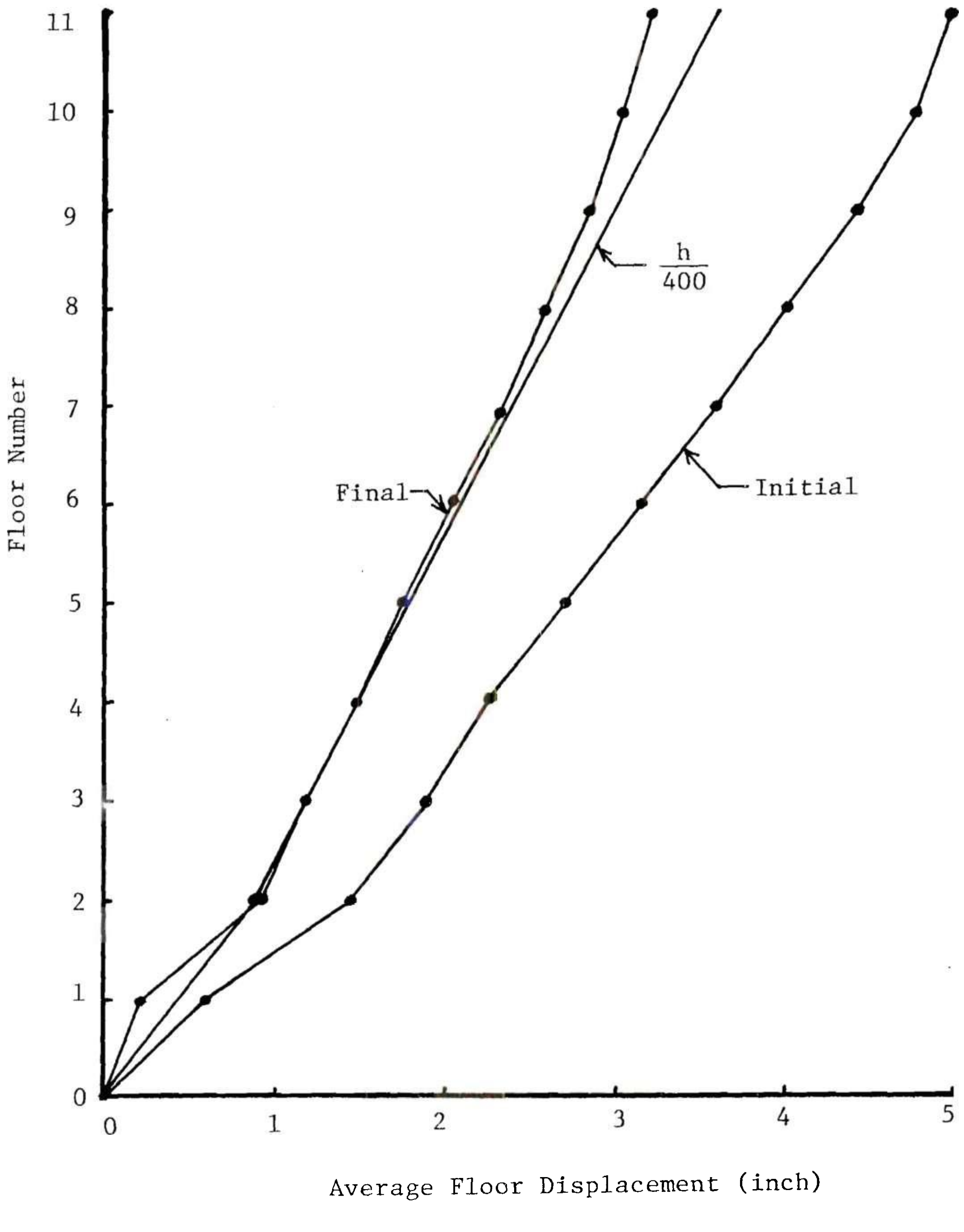


Figure 5.25 Average Floor Displacement, Z-Direction - Example B1 (EP = 25%)

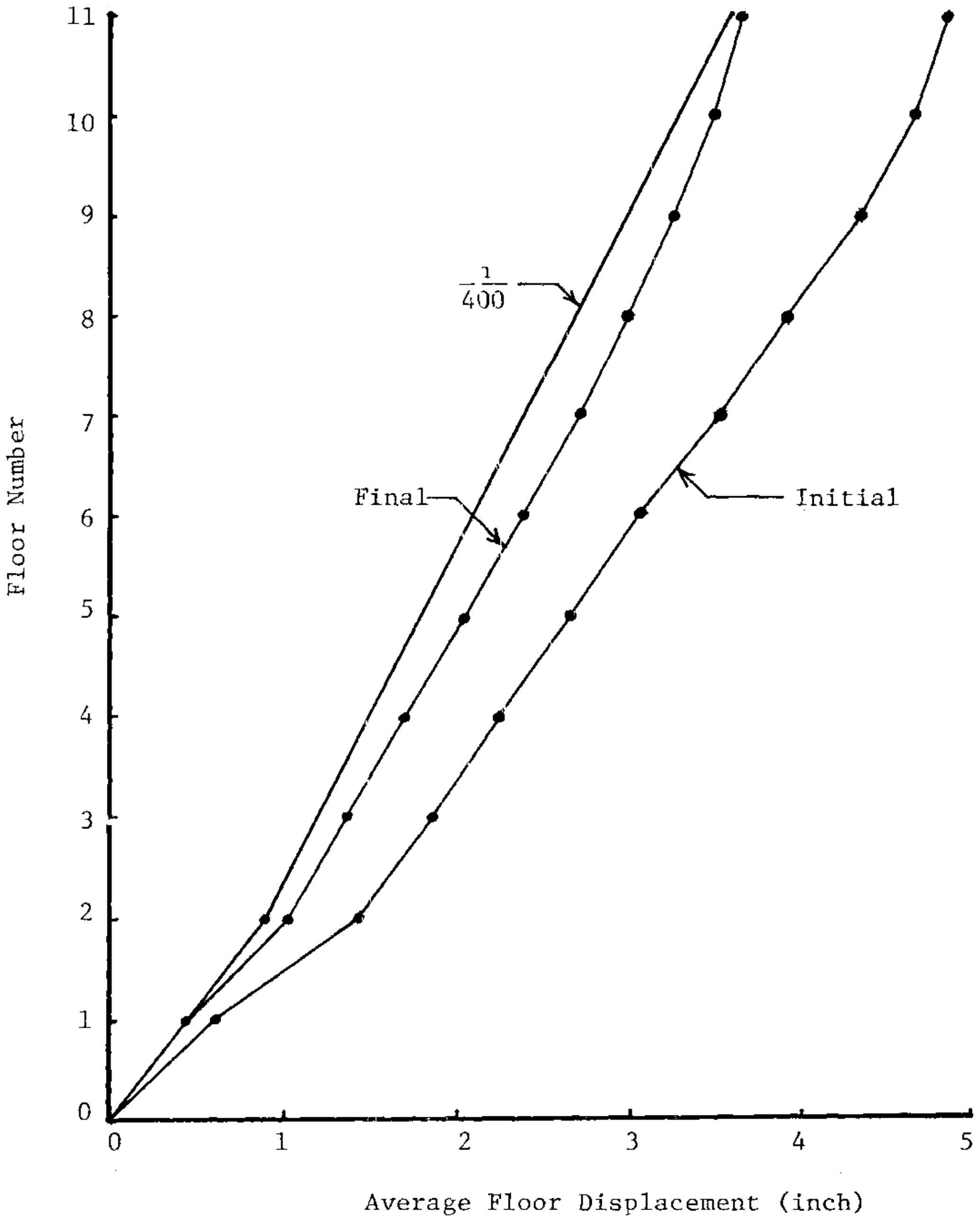


Figure 5.26 Average Floor Displacement, X-Direction - Example B2 (EP = 10%)

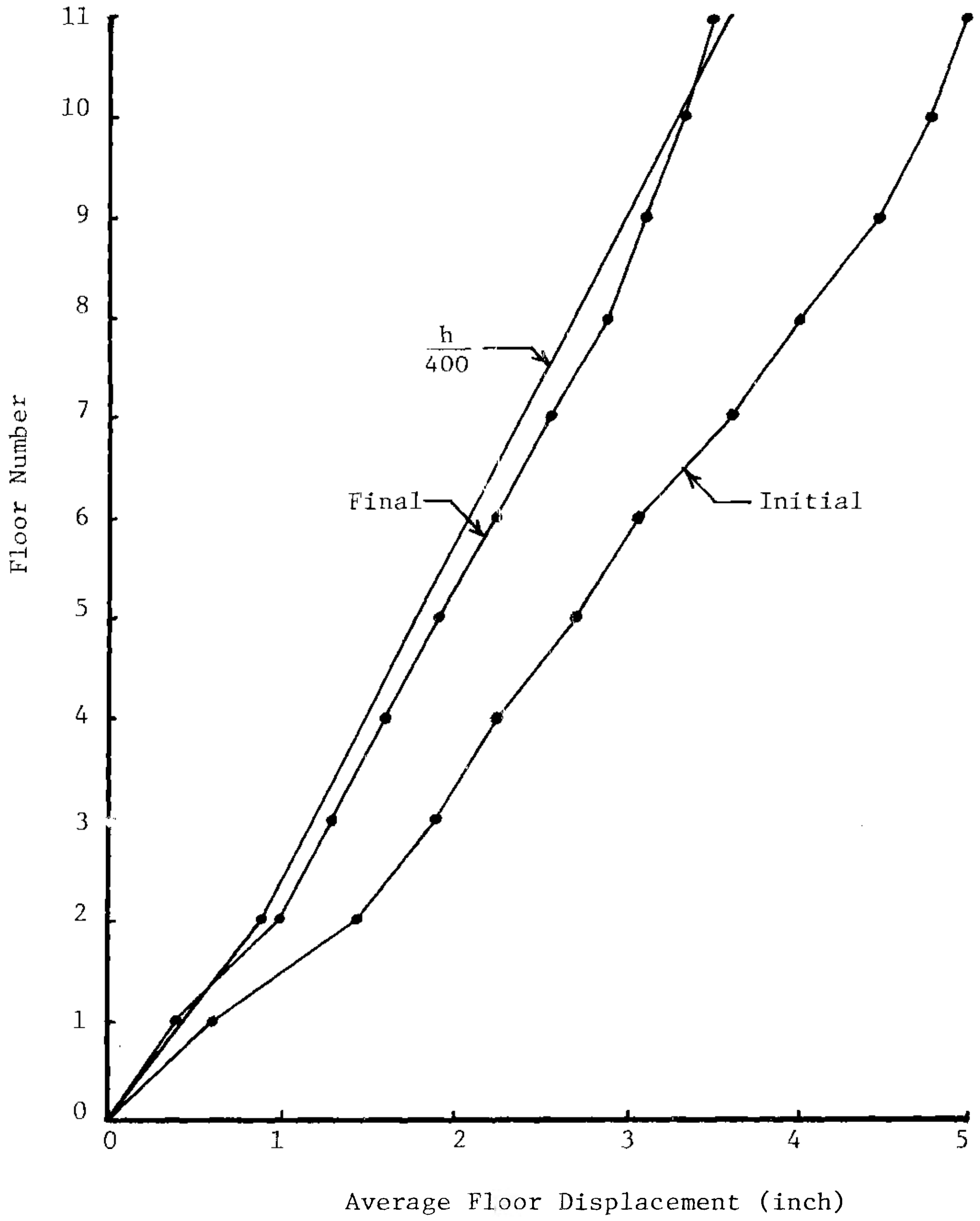


Figure 5.27 Average Floor Displacement, Z-Direction - Example B2 (EP = 10%)

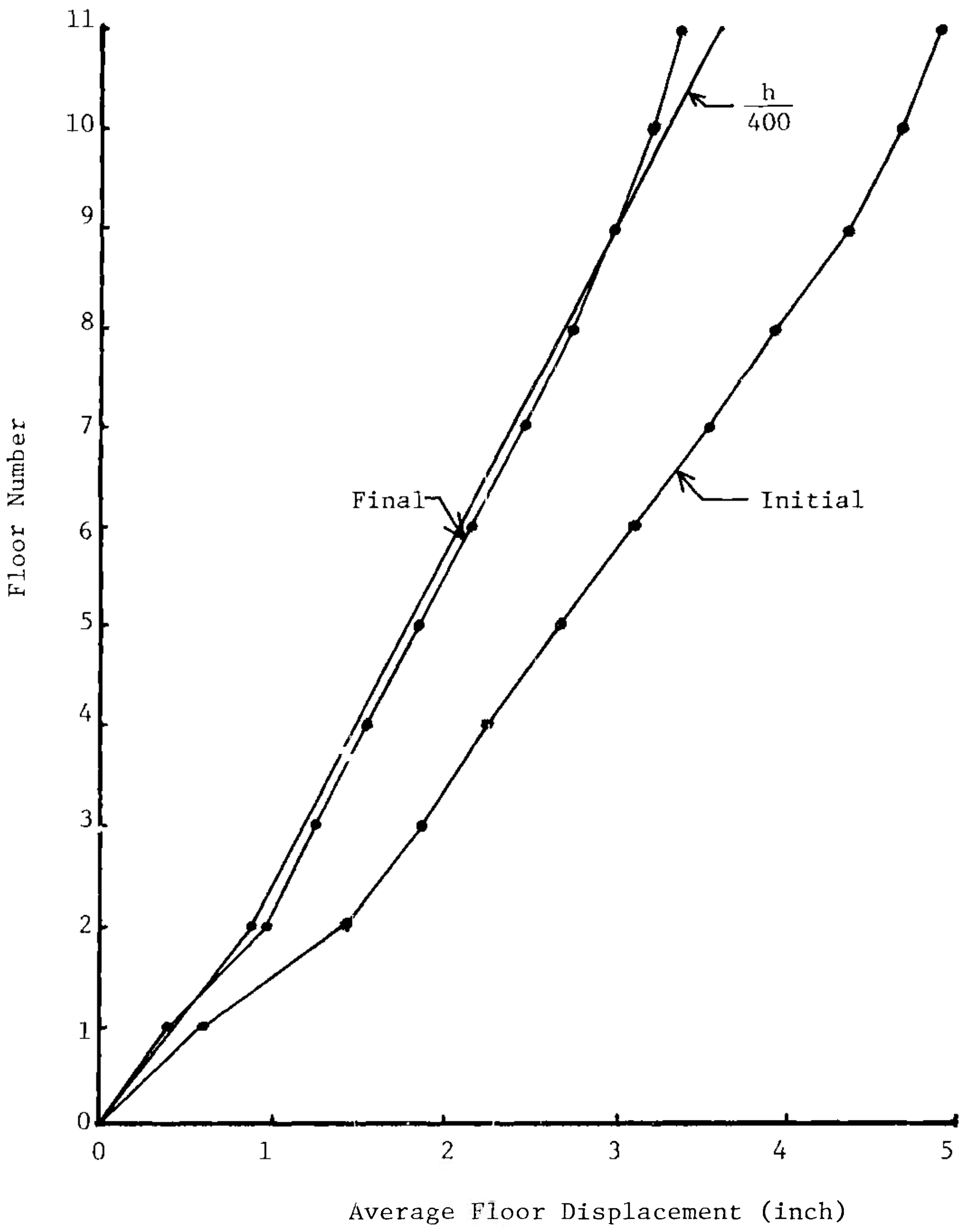


Figure 5.28 Average Floor Displacement, X-Direction - Example B3 (EP = 0%)

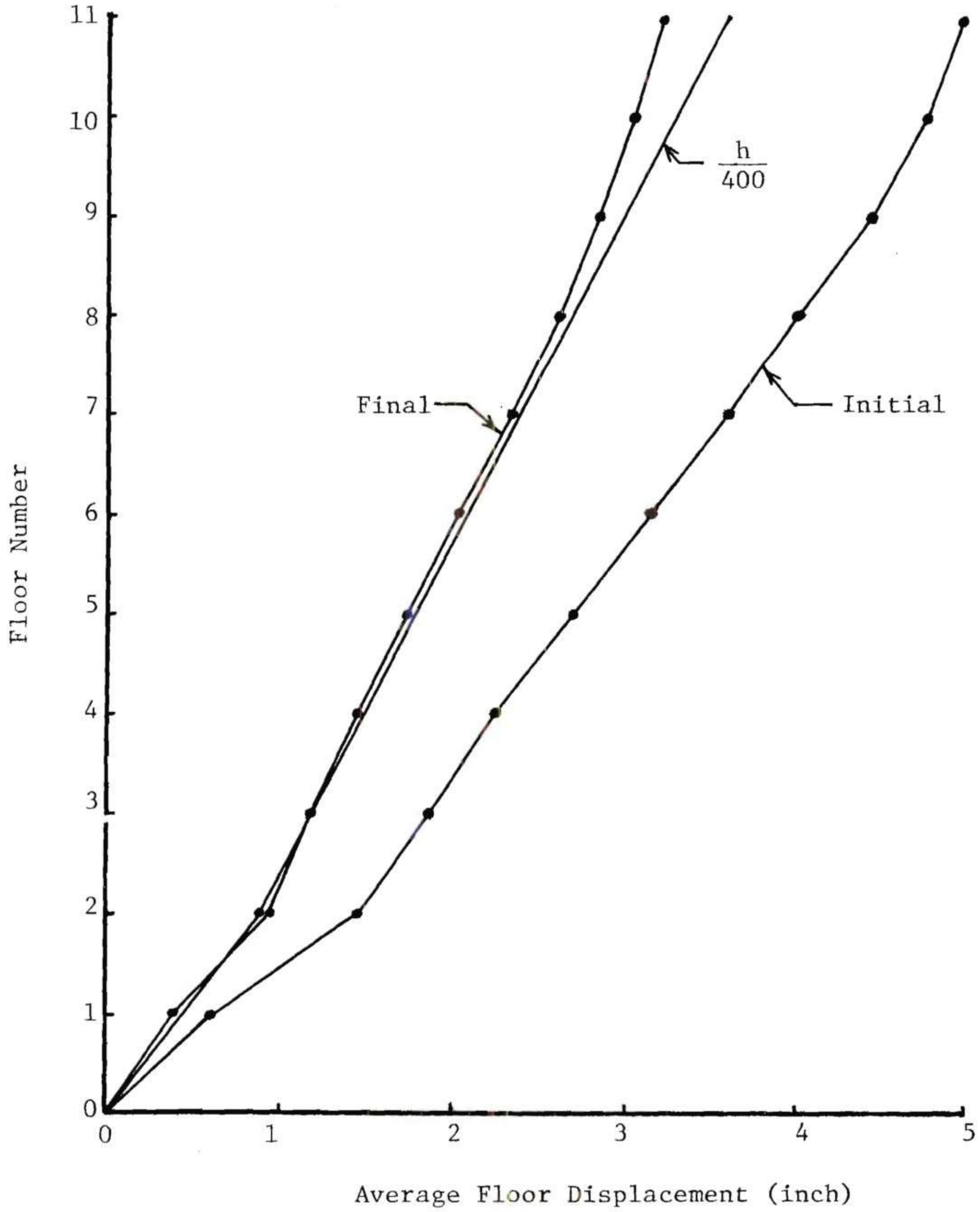


Figure 5.29 Average Floor Displacement, Z-Direction - Example B3 (EP = 0%)

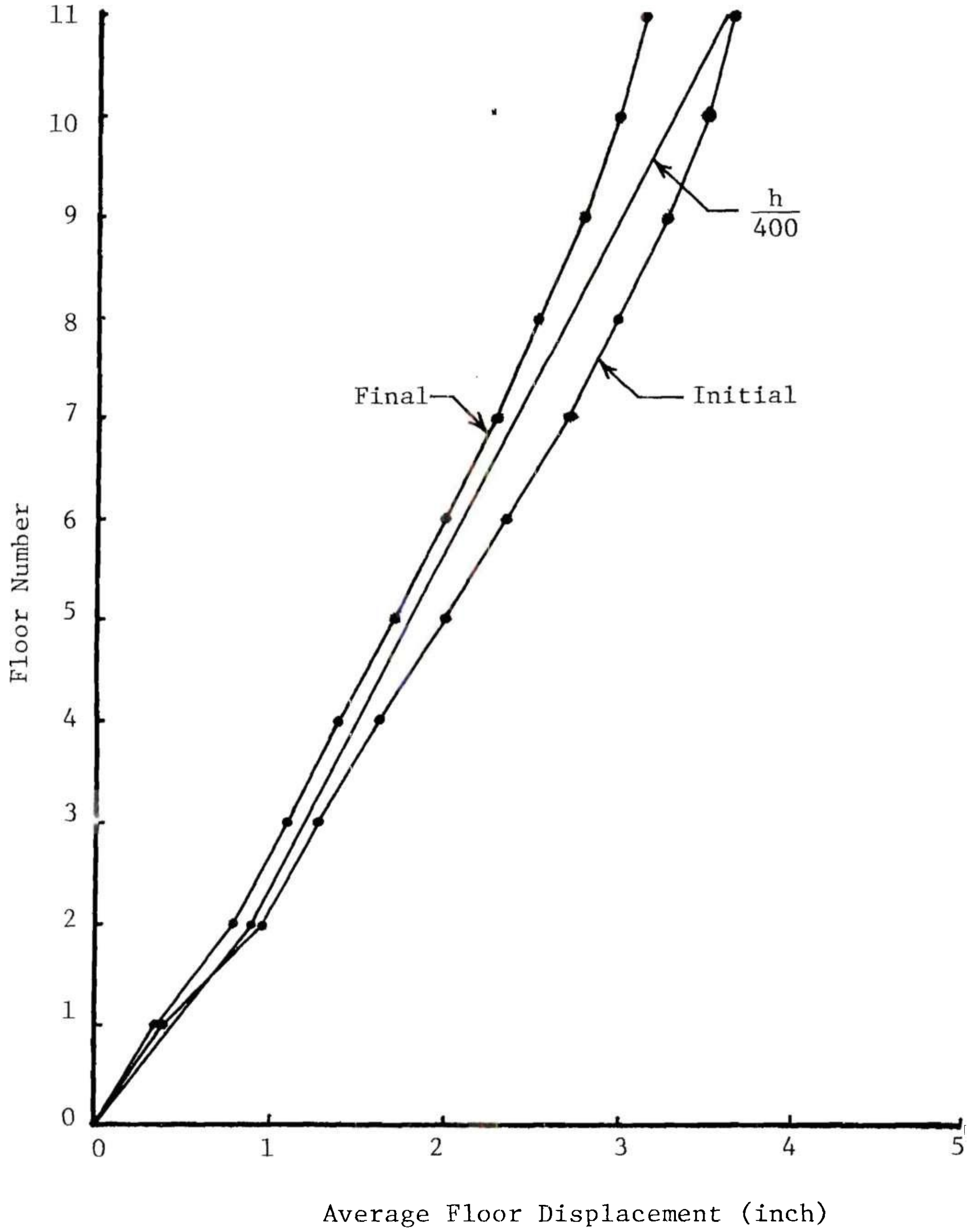


Figure 5.30 Average Floor Displacement, X-Direction - Example BW2 (EP = 10%)

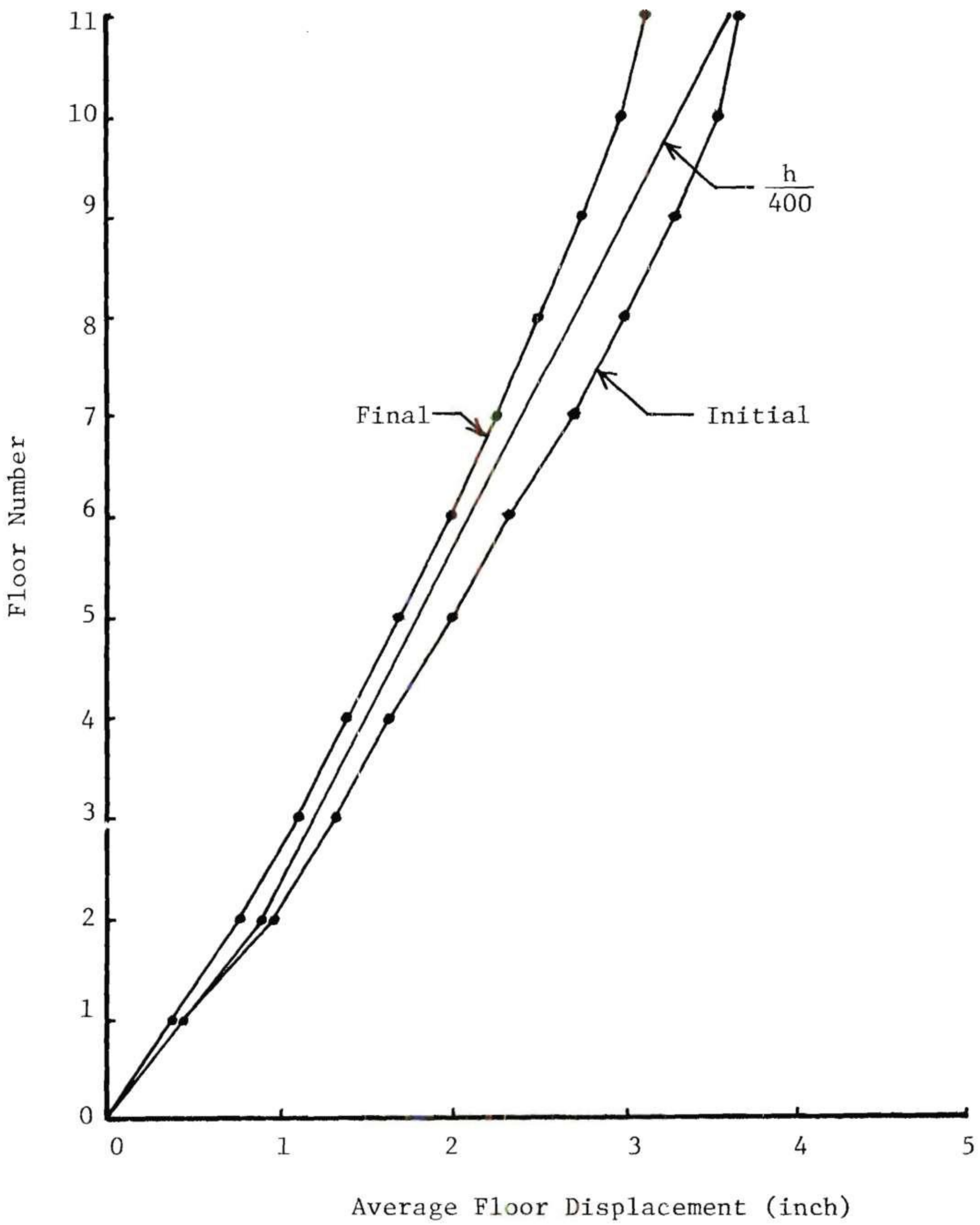


Figure 5.31 Average Floor Displacement, Z-Direction - Example BW2 (EP = 10%)

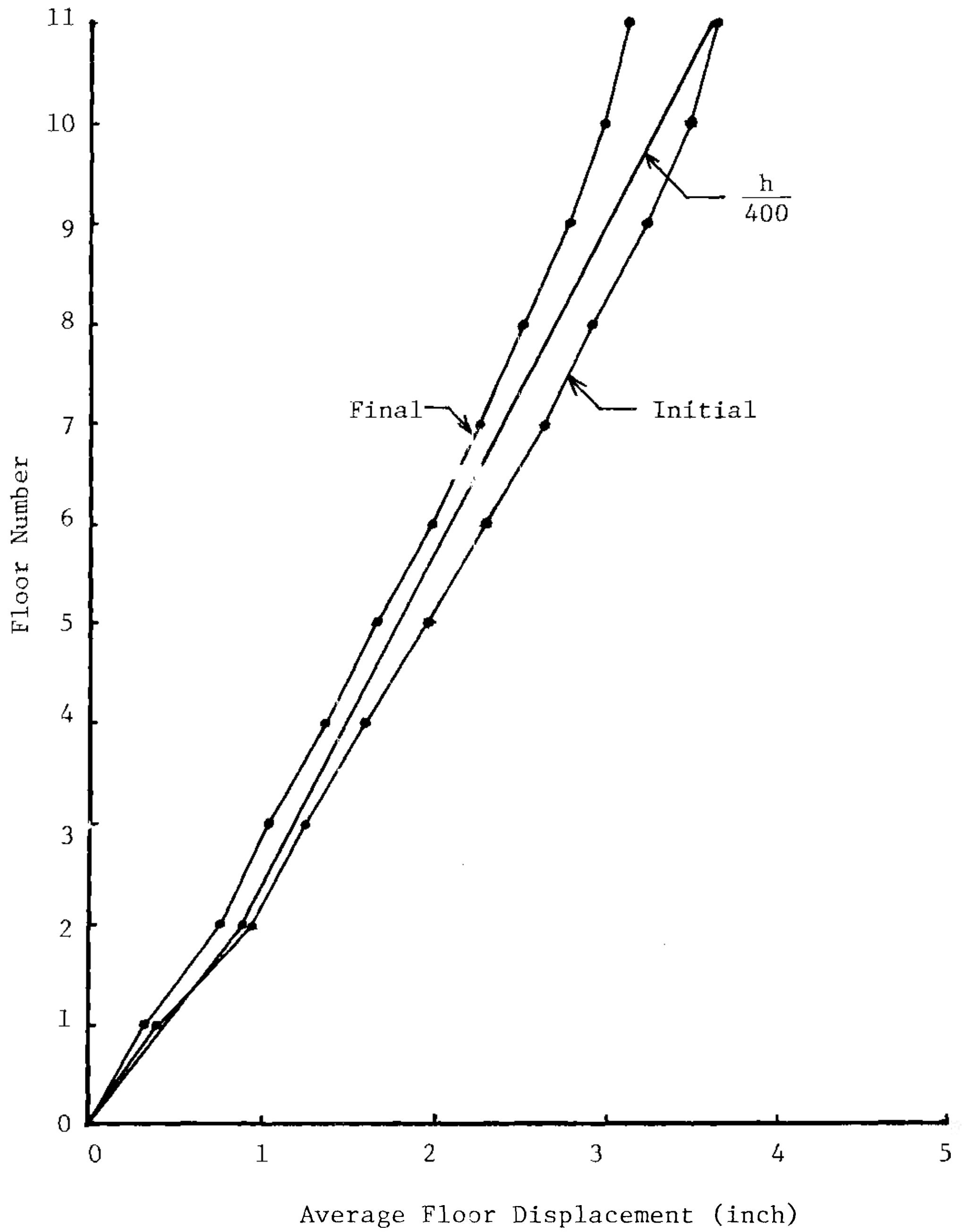


Figure 5.32 Average Floor Displacement, X-Direction - Example BRW2 (EP = 10%)

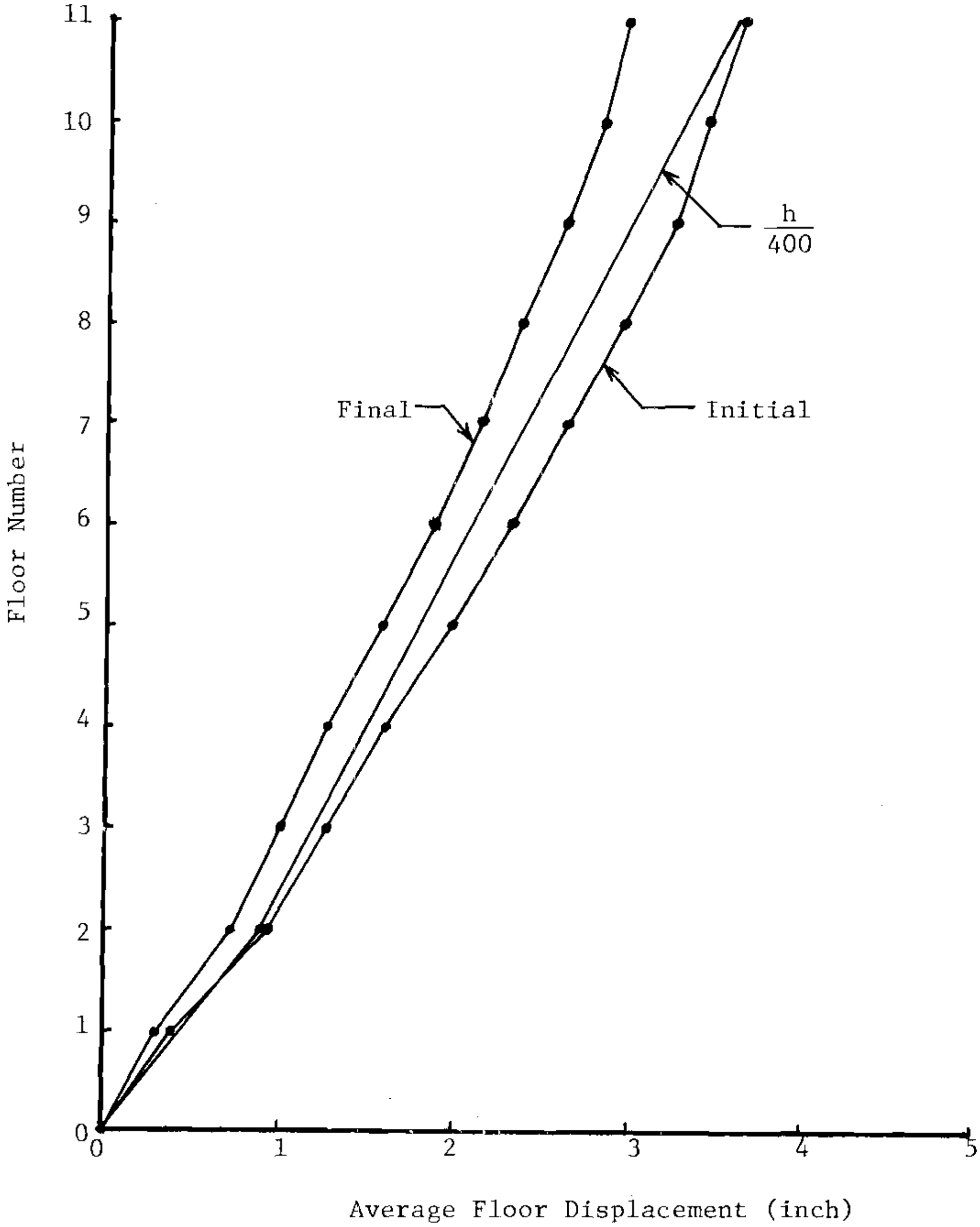


Figure 5.33 Average Floor Displacement, Z-Direction - Example BRW2 (EP = 10%)

the amount of computer time required to perform a full scale analysis. It was developed to serve as a middle level displacement approximation, not as time consuming and accurate as an exact stiffness analysis, but not as approximate as the virtual work procedure. Upon testing the method, it was found to be extremely time consuming and cumbersome, so much so, that it was many times slower than performing an exact stiffness analysis! Thus, the decision was made to bypass its use in all Frame A and B design examples.

Several factors can be cited as contributing to the apparent inefficiency of the kinematic condensation procedure in the context of this static analysis and stiffness design computer system. These include the necessity of performing a large number of matrix multiplications (Eq.3.8), the requirement to extensively utilize secondary storage for intermediate information storage, and most importantly, the fact that the final reduced stiffness equations (Eq.3.8) do not exhibit any banded properties.

To illustrate results obtained from the kinematic condensation procedure, it was applied to the following structure: 2 story, 1 bay by 1 bay, space frame shown in Fig. 5.34, with member sizes shown in Table 5.13, and subjected to a 10 kip load in the global X-direction at joint 9. Three rigid body displacements per floor, X-displacement, Z-displacement and Y-rotation, exist for the case of floors having in-plane rigidity, and these are shown in Table 5.14 (a), with corresponding joint displacements shown in Table 5.16. Six rigid body deflections per story, X, Y, and Z translations plus X, Y, and Z rotations are computed when a fully rigid floor is assumed, and these are shown in Table 5.14 (b), with all corresponding joint displacements shown in Table 5.17. Exact displacements computed from an exact stiffness analysis are presented in Table 5.15 for the purpose of comparison. Rigid body displacements, referred to as displacement measures in Chapter 3, refer to the movement of points designated as the center of the diaphragms being modeled. Examples of such points are P_k and P_{k+1} in Fig. 3.2, and P_1 and P_2 in Fig. 5.34. Dependent deflections at joints on each floor are related to corresponding rigid body deflections at these points (P_1 and P_2) as described in Chapter 3.

Joint 9, from the structure in Fig. 5.34, is chosen to explain the calculation of dependent displacements at a joint from the values of the displacement measures for the floor on which it is located. First consider the situation of in-plane floor rigidity. Eqs. 3.8, 3.9 and 3.10 describe the motion of a joint on such a diaphragm. Table 5.14(a) shows the values of d_{kx}, d_{kz}, and r_{ky} to be .584, $-4.33\text{x}10^{-5}$ and $3.59\text{x}10^{-3}$, respectively for the second floor. Center point P_2 has an X coordinate of 120 and Z coordinate of 120 as indicated in Fig. 5.34. The Y coordinate is not required since all joints on a floor are taken to be in a horizontal plane. Quantities e_{jx} and e_{jz} in Eqs. 3.8, 3.9 and 3.10 are both equal to 120. With this information, the following dependent displacements result at joint 9 (Eqs. 3.8, 3.9 and 3.10),

$$u_{91} = .584 - 120\ (-.00359) = 1.0148 \text{ (X displ.)}$$

$$u_{93} = -4.33 \times 10^{-5} + 120\ (-.00359) = -.43084 \text{ (Z displ.)}$$

$$u_{95} = -.00359 \text{ (Y rotation)}$$

All such dependent displacements, as well as the independent displacements are shown in Table 5.16.

Eqs. 3.19 to 3.24 describe the motion of a joint on a floor which is assumed to behave as a completely rigid diaphragm. For this condition, all displacements are dependent. As before, consider joint 9. Table 5.14 (b) shows the displacement measures d_{kx}, d_{ky}, d_{kz}, r_{kx}, r_{ky}, and r_{kz}, which equal .305, $-3.30\text{x}10^{-14}$, $-2.08\text{x}10^{-5}$, $-3.24\text{x}10^{-10}$, $-2.05\text{x}10^{-3}$, $-2.44\text{x}10^{-5}$, for point P_2 on the top floor. Dependent displacements are computed as follows (Eqs. 3.19 to 3.24),

$$u_{91} = .305 + (-120)(-.00205) = .551 \text{ (X displ.)}$$

$$u_{92} = -3.30\text{x}10^{-14} - (-120)(-3.24\text{x}10^{-10}) + (-120)(-2.44\text{x}10^{-5})$$

$$= .00293 \text{ (Y displ.)}$$

$$u_{93} = -2.08\text{x}10^{-5} - (-120)(-.00205) = -.246 \text{ (Z displ.)}$$

$$u_{94} = -3.24\text{x}10^{-10} \text{ (X rotation)}$$

$$u_{95} = -.00205 \text{ (Y rotation)}$$

$$u_{96} = -2.44\text{x}10^{-5} \text{ (Z rotation)}$$

All such dependent displacements are shown in Table 5.17.

The results obtained from the kinematic condensation analysis for in-plane floor rigidity are very good. The close comparison between exact displacements in Table 5.15 and kinematic condensation displacements of in-plane rigid floors in Table 5.16 show that this space frame behaves like one with in-plane floor rigidity, rather than complete floor rigidity. Very poor comparison results are shown between exact displacements in Table 5.15, and completely rigid floor kinematic condensation displacements in Table 5.17. This occurs simply because the frame is much more accurately modeled as having in-plane rigid floors.

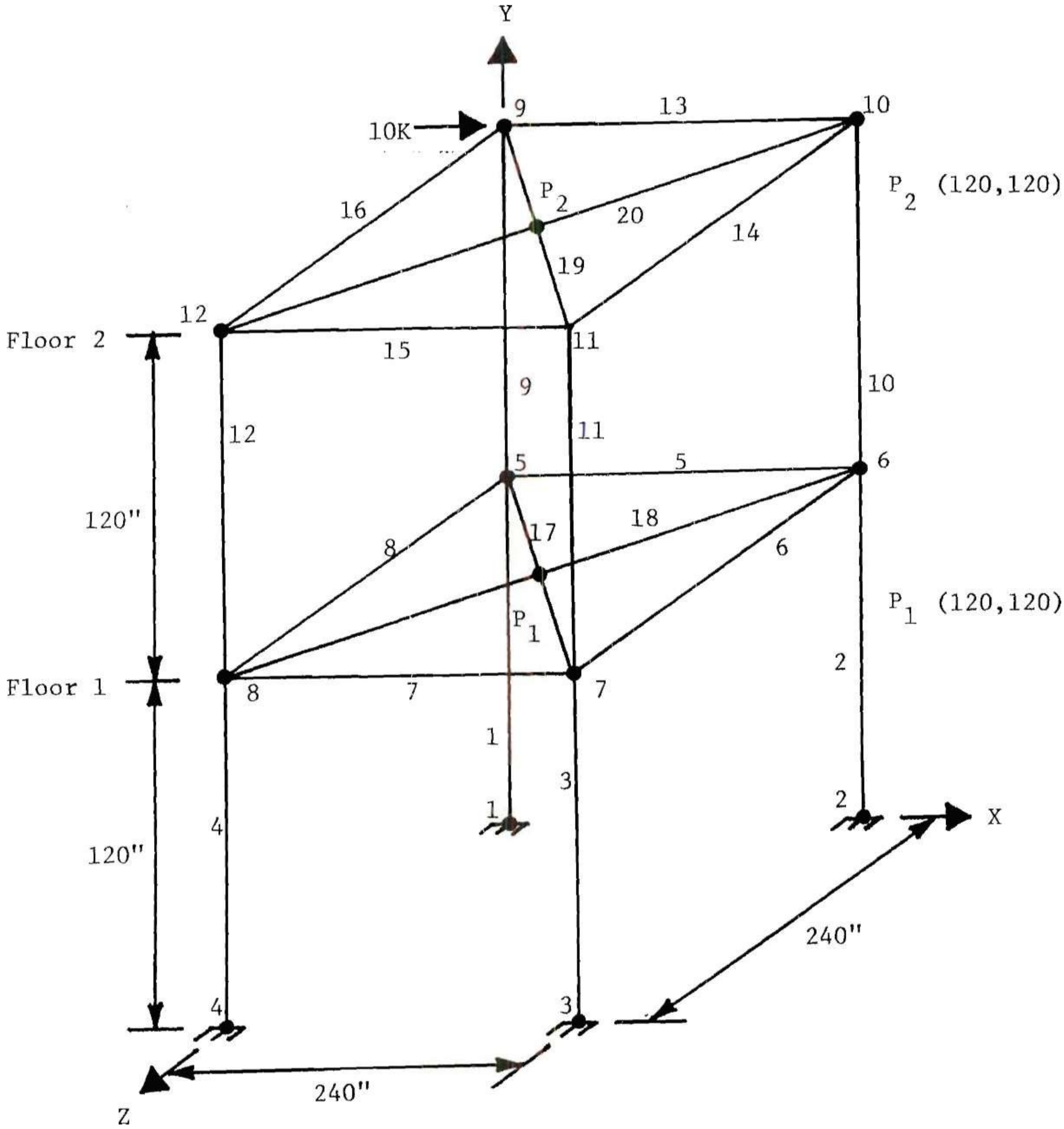


Figure 5.34 Kinematic Condensation Analysis Example

Table 5.13. Member Incidences and Properties for Example Structure of Figure 5.34

Member Number	Start Joint	End Joint	Member Section Size
1	1	5	8WF24
2	2	6	do
3	3	7	do
4	4	8	do
5	5	6	14B22
6	6	7	do
7	7	8	do
8	5	8	do
9	5	9	8WF24
10	6	10	do
11	7	11	do
12	8	12	do
13	9	10	14B22
14	10	11	do
15	11	12	do
16	9	12	do
17	5	7	3UAN9.0
18	6	8	do
19	9	11	do
20	10	12	do

Table 5.14 Displacement Measures for Example Structure of Figure 5.34

a. In-Plane Floor Rigidity (units = inch and radians)

	Floor 1	Floor 2
X-displacement:	0.257	0.584
Z-displacement:	-2.42×10^{-5}	-4.33×10^{-5}
Y-rotation	-0.00161	-0.00359

b. Complete Floor Rigidity (units = inch and radians)

	Floor 1	Floor 2
X-displacement:	0.152	0.305
Y-displacement:	-2.12×10^{-14}	-3.30×10^{-14}
Z-displacement:	-1.25×10^{-5}	-2.08×10^{-5}
X-rotation:	-2.53×10^{-10}	-3.24×10^{-10}
Y-rotation:	-1.03×10^{-3}	-2.05×10^{-3}
Z-rotation:	-1.83×10^{-5}	-2.44×10^{-5}

Table 5.15 Exact Displacements for Structure of Figure 5.34 (inches)

Joint	X-displacement	Y-displacement	Z-displacement	X-rotation	Y-rotation	Z-rotation
1	0	0	0	0	0	0
2	0	0	0	0	0	0
3	0	0	0	0	0	0
4	0	0	0	0	0	0
5	.451	.00295	-.194	-5.09×10^{-4}	-.00161	-3.10×10^{-3}
6	.451	-.00295	.194	5.08×10^{-4}	-.00161	-3.08×10^{-3}
7	.0638	-.00110	.194	5.09×10^{-4}	-.00161	-4.64×10^{-4}
8	.0638	.00110	-.194	-5.08×10^{-4}	-.00161	-4.63×10^{-4}
9	1.018	.00411	-.429	-2.85×10^{-4}	-.00359	-2.01×10^{-3}
10	1.018	-.00411	.428	2.83×10^{-4}	-.00357	-1.98×10^{-3}
11	.152	-.00152	.428	2.84×10^{-4}	-.00357	-3.22×10^{-4}
12	.152	.00152	-.429	-2.83×10^{-4}	-.00359	-3.19×10^{-4}

Table 5.16 Displacements - In-Plane Floor Rigidity for Structure of Figure 5.34 (inches)

Joint	X-displacement	Y-displacement	Z-displacement	X-rotation	Y-rotation	Z-rotation
1	0	0	0	0	0	0
2	0	0	0	0	0	0
3	0	0	0	0	0	0
4	0	0	0	0	0	0
5	.451	.00294	-.193	-5.11×10^{-4}	-.00161	-3.09×10^{-3}
6	.451	-.00294	.193	5.11×10^{-4}	-.00161	-3.09×10^{-3}
7	.0638	-.00110	.193	5.11×10^{-4}	-.00161	-4.66×10^{-4}
8	.0638	.00110	-.193	-5.11×10^{-4}	-.00161	-4.66×10^{-4}
9	1.015	.00410	-.431	-2.88×10^{-4}	-.00359	-1.99×10^{-3}
10	1.015	-.00410	.431	2.88×10^{-4}	-.00359	-1.99×10^{-3}
11	.153	-.00153	.431	2.88×10^{-4}	-.00359	-3.24×10^{-4}
12	.153	.00153	-.431	-2.88×10^{-4}	-.00359	-3.24×10^{-4}

Table 5.17 Displacements - Complete Floor Rigidity for Structure of Figure 5.34 (inches)

Joint	X-displacement	Y-displacement	Z-displacement	X-rotation	Y-rotation	Z-rotation
1	0	0	0	0	0	0
2	0	0	0	0	0	0
3	0	0	0	0	0	0
4	0	0	0	0	0	0
5	.275	.00220	-.123	-2.54×10^{-10}	-.00103	-1.83×10^{-5}
6	.275	-.00220	.123	-2.54×10^{-10}	-.00103	-1.83×10^{-5}
7	.0282	-.00220	.123	-2.54×10^{-10}	-.00103	-1.83×10^{-5}
8	.0282	.00220	-.123	-2.54×10^{-10}	-.00103	-1.83×10^{-5}
9	.551	.00293	-.247	-3.24×10^{-10}	-.00205	-2.44×10^{-5}
10	.551	-.00293	.247	-3.24×10^{-10}	-.00205	-2.44×10^{-5}
11	.0580	-.00293	.247	-3.24×10^{-10}	-.00205	-2.44×10^{-5}
12	.0580	.00293	-.247	-3.24×10^{-10}	-.00205	-2.44×10^{-5}

Complete generality is an advantageous feature of the kinematic condensation process described herein. The method is structure independent, since the user is free to make any assumptions concerning displacement relationships which he deems valid by either inputting a $\underset{\sim}{T}$ matrix, or taking advantage of one of the conditions which is internally programmed (in-plane or complete floor rigidity). Unfortunately, the disadvantage for use in this design system is the amount of time necessary to execute the kinematic condensation. For the structure of Fig. 5.34, the time required to perform the exact stiffness analysis was a little more than 7 CPU seconds on an IBM 370/158 computer. Each of the kinematically condensed models required approximately 110 CPU seconds. Hence, it is clear why kinematic condensation is not used in the design examples and is inefficient relative to the purpose for which it was intended. Repeated analyses, which may be required would be too costly. Consequently, for use in this stiffness design system, kinematic condensation analysis as described herein, is not justified.

CHAPTER 6

CONCLUSIONS AND RECOMMENDATIONS

Several important conclusions emerged from this study. They are:

1. The stiffness design system is highly cost effective (as long as the kinematic condensation option is not requested), producing final frame designs which satisfy stringent displacement constraints. For a typical Example Frame B design (110 beam, 88 column, 22 brace, and 96 joint space frame), the computer time cost was under $30. Since the computer design selects members on a member-by-member basis, and does not adjust such sizes for fabrication or erection economy, the resulting final design must be used by the structural engineer as a rational guideline to produce a final design for construction purposes.
2. The generalized kinematic condensation procedures developed herein, and implemented, were not sufficiently efficient to be recommended for further use in this computer design system. However, the generality of the procedure is worth considering for use in large scale dynamic analysis programs.
3. Although the displacement criterion is satisfied at each displacement constraint point, it may not be satisfied at other points. In the unbraced frame examples in this study, it was found that when displacement constraints are placed only on

the top floor, the floors near the mid-height of the building frame violated the displacement criterion (e.g., h/400) by a small amount. However, when a constraint point was located near the mid-height of the frame, all floors satisfied the criterion. It is recommended, therefore, that for unbraced frames, at least one displacement constraint location should be specified in the middle third of the frame, in addition to the top floor. In the braced frame examples, however, this was not a problem. For braced frames, it is recommended that displacement constraints need to be located only on the top floor.

4. In regard to member size change trends, for the unbraced 24-story plane frame (Example A2), almost all (96%) of the mem-size changes occurred in the beams, while only about 4% occurred in the columns. This was as expected. For the braced 24-story plane frame (Example AWR2), an unexpected trend was observed in that about 76% of the member size changes (number of changes) occurred in the beams, while only 20% occurred in the braces, and 4% in the columns. One would have expected a larger proportion of changes to have occurred in the braces, but that was not the case.

   On the other hand, for the unbraced 11-story space frame (Example B2), about 55% of all member size changes occurred in the beams, and 45% occurred in the columns. This was not surprising. For the braced 11-story space frame (Example BW2), about 28% of all member size changes occurred in the beams, 12% in the columns, and 60% in the braces. This was also expected.

5. It was found that a user specified initial error term of 10% of the starting exact displacement at constraint locations leads to the most reasonable final designs for the types of frames considered. In addition, for the 10% initial error term, the last (critical) deflection constraint to be satisfied in the example problems considered was reduced below the constraint value in the range 0.3% - 8.8% with an average of 2.38%. This is considered to be very good.

It is recommended that future extensions and enhancements to this design system consider the following:

1. Implement the design system in a more comprehensive structural analysis and design system, such as ICES STRUDL II (7,8), so that the benefits of this work may be made available in the fastest possible way.
2. Enhance the design system to: (a) permit geometric constraints on member properties such as member depth limitations, (b) add braces to a frame where there is some flexibility in this regard, (c) permit consideration of relative story displacement constraints, (d) permit at least six or more displacement constraints to be specified, (e) permit similarity, constraints to be specified for different groups of members in the frame (i.e., two or more story column lengths, all beams in a floor, or multiple floors, to be the same, etc.), (f) permit member size changes to decrease a member size, in addition

to increasing its size, (g) permit individual joint rotation constraints to be specified, and (h) permit kinematic condensation to treat plane frames as plane frames.

3. Improve the efficienty of the middle level, approximate, generalized kinematic condensation displacement analysis, or implement a new approach such as it described by Weaver, et al (16,17,18).

REFERENCES

1. American Institute of Steel Construction, Inc., Manual of Steel Construction, AISC, 1970.

2. Brown, Daniel M., and Ang, Alfredo, H.S., Structural Optimization by Nonlinear Programming, Journal of the Structural Division, ASCE, VOL. 92, No. ST6, Proc. Paper 5016, December, 1966, pp. 319-340.

3. Cella, Aldo, and Logcher, Robert D., Automated Optimum Design From Discrete Components, Journal of the Structural Division, ASCE, VOL. 97, No. ST1, Proc. Paper 7845, January, 1971, pp. 175-189.

4. Chen, Peter W., and Robertson, Leslie E., Human Perception Thresholds of Horizontal Motion, Journal of the Structural Division, ASCE, VOL. 98, No. ST8, Proc. Paper 9142, August, 1972, pp. 1681-1695.

5. Feming, J.F., Podolny, W., Jr., and Rile, T.S., Drift Study of Tapered Frameworks, ASCE Civil Engineering, August, 1967.

6. Gere, James M., and Weaver, William, Jr., Analysis of Framed Structures, D. Van Nostrand Company, Inc., Princeton, N.J., 1965.

7. ICES STRUDL II Engineering User's Manual, Volume 1, Frame Analysis, Massachusetts Institute of Technology, Department of Civil Engineering Research Report R68-91, November, 1968.

8. ICES STRUDL II Engineering User's Manual, Volume 2, Additional Design and Analysis Facilities, Second Edition, Massachusetts Institute of Technology, Department of Civil Engineering Research Report R70-77, June, 1971.

9. Kavlie, Dag, and Moe, Automated Design of Framed Structures, Journal of the Structural Division, ASCE, Vol. 97, ST1, Proc. Paper 7823, January, 1971, pp. 33-62.

10. LeMessurier, W.J., Summary Report, Technical Committee No. 17, Design Methods Based on Stiffness, Proceedings of the International Conference on Planning and Design of Tall Buildings, Vol. II, ASCE, August, 1972.

11. Moses, Fred, Optimum Structural Design Using Linear Programming, Journal of the Structural Division, ASCE, Vol. 90, No. ST6, Proc. Paper 4163, December, 1964.

12. Reed, John W., Wind-Induced Motion and Human Discomfort in Tall Buildings, M.I.T. Department of Civil Engineering Research Report No. R71-42, November, 1971.

13. Romstad, Karl M., and Wang, Chu-Kia, Optimum Design of Framed Structures, Journal of the Structural Division, ASCE, Vol. 94, No. ST12, Proc. Paper 6273, December, 1968, pp. 2817-2845.

14. Scalzi, J.B., Drift in High Rise Steel Framing, Progressive Architecture, Reinhold Publications, April, 1972.

15. Weaver, William, Jr., Computer Programs for Structural Analysis, D. Van Nostrand Company, Inc., Princeton, N.J., 1967.

16. Weaver, William, Jr., and Nelson, Mark F., Three Dimensional Analysis of Tier Buildings, Journal of the Structural Division, ASCE, Vol. 92, No. ST6, Proc. Paper 5019, December, 1966, pp. 385-404.

17. Weaver, William, Jr., and Brandow, Gregg E., Tier Buildings with Shear Cores, Bracing, and Setbacks, Journal of Computers and Structures, Pergamon Press, Vol. 1, 1971, pp. 57-83.

18. Weaver, William, Jr., Nelson, Mark F., and Manning, Thomas A., Dynamics of Tier Buildings, Journal of the Engineering Mechanics Division, ASCE, Vol. 94, No. EM6, Proc. Paper 6293, December, 1968, pp. 1455-1473.

APPENDIX A

PROGRAM INPUT

The following is a presentation of the input in the order in which it is required to appear for proper execution of the design system. It is assumed the reader is familiar with the FORTRAN IV READ and FORMAT statements. Note that limitations on the size of problem that may be treated by this system in its present form, which are not specifically covered by the input descriptions in this Appendix, are enumerated in Appendix F along with the current program listing.

All units for input data must be inches and kips except where noted.

1. READ (TITLE(I),I=1,20)

   FORMAT (20A4)

   One data card where,

   TITLE = A maximum of 80 alphanumeric characters which identify the current problem.

2. READ ICOL,IBEAM,IBR,IEXACT,ITOTAL,ITRANS,IUPD,MEMCHG,IFLAG, IFLAG1,NLS,NDC,FACT,TOL,EP

   FORMAT (12I5,F8.2,2F6.2)

   One data card where,

   ICOL = Number of column sections to be input.

   IBEAM = Number of beam sections to be input.

   IBR = Number of bracing sections to be input.

   IEXACT = Number of kinematic condensation analyses per exact analysis.

   ITOTAL = Maximum number of analyses (stiffness analyses and kinematic condensation analyses) permitted.

   ITRANS = Transformation matrix ($\underset{\sim}{T}$) parameter.

   = 0 = Transformation matrix ($\underset{\sim}{T}$) is to be input.

   = 1 = Transformation matrix ($\underset{\sim}{T}$) for completely rigid floor is to be formed internally by the program.

   = 2 = Transformation matrix ($\underset{\sim}{T}$) for in-plane floor rigidity is to be formed internally by the program.

   IUPD = Maximum number of total member size changes per stiffness or kinematic condensation analysis.

MEMCHG = Member change parameter.

= 0 = Change only that member with the most negative displacement sensitivity coefficient (regardless of which constraint it is associated with, if multiple constraints exist) before checking updated displacements against displacement constraint values.

= 1 = Permit one member size change to be associated with each displacement constraint (i.e. that member with the most negative displacement sensitivity coefficient for each constraint) except under special circumstances (see Section 2.5).

TFLAG = Initial analysis type parameter.

= 0 = Perform an exact stiffness analysis as the first analysis.

= 1 = Perform a kinematic condensation analysis as the first analysis.

TFLAGL = Final analysis type parameter.

= 0 = Perform an exact stiffness analysis as the last analysis.

= 1 = Perform a kinematic condensation analysis as the last analysis.

NLS = Total number of loading conditions (applied lateral loads + virtual loads). ($1 \leq NLS \leq 7$)

NDC = Number of displacement constraints ($1 \leq NDC \leq 3$).

FACT = Unit load magnification factor. (If the magnitude of the virtual force corresponding to a displacement constraint, a Q load, is larger than unity, FACT equals the magnitude of the applied virtual force. Note that all virtual forces must be of equal magnitude as FACT is used to modify displacements and member forces resulting from all virtual loads.)

TOL = Amount by which a computed displacement may exceed a displacement constraint expressed as a percentage of the displacement constraint (e.g. 3.0 for 3%).

EP = Initial error, expressed as a percent of the actual displacement calculated in the first analysis and corresponding to the displacement constraint locations.

- -

Steps 3 to 6 are repeated for each displacement constraint, a total of NDC times (I=1,NDC).

- -

3. READ IQPT(I,J),J=1,6)

FORMAT (6I5)

One data card where,

IQPT(I,1) = Loading condition number of the virtual loading (Q load) associated with displacement constraint I.

IQPT(I,2) = Number of applied lateral loading conditions (P loads) for which displacement constraint I must be satisfied.

IQPT(I,3) = Loading condition number of first applied loading condition (P load) for which displacement constraint I must be satisfied.

IQPT(I,4) = Loading condition number of second applied loading condition (P load) for which displacement constraint I must be satisfied.

IQPT(I,5) = Loading condition number of third applied loading condition (P load) for which displacement constraint I must be satisfied.

IQPT(I,6) = Loading condition number of fourth applied loading condition (P load) for which displacement constraint I must be satisfied.

A maximum of four applied loading conditions may be required to satisfy displacement constraint I. If less than four are desired, the remaining values of IQPT(I,J) are equal to zero.

4. READ (JT(I,J),J=1,2)

FORMAT (2I5)

This input is dependent on the type of displacement constraint I, that is, whether it is rotational or translational. For a translational constraint, (IDTYP(I)=1, from Input Step 6), the following applies.

One data card where,

JT(I,1) = Joint number at which a displacement constraint is imposed.

JT(I,2) = 0

For a rotational constraint (IDTYP(I)=0, from Input Step 6), the following applies.

One data card where,

JT(I,1) = One joint number specified on a floor where rotation is to be controlled.

JT(I,2) = A second joint number on the same floor where rotation is to be controlled.

For a rotational constraint, these two joints define a chord in the floor and the rotation of the floor is to be determined by the rotation of this chord.

5. READ (XJT(I,J),J=1,4)

FORMAT (4F10.2)

This input is dependent on the type of displacement constraint under consideration. First, if a translational constraint is being input (IDTYP(I)=1, from Input Step 6), the array should be of the following form.

One data card where,

XJT(I,1) = Global X component of a unit vector which is in the direction for which the displacement constraint I is imposed at the designated joint (JT(J,1)).

XJT(I,2) = Global Y component of a unit vector which is in the direction for which the displacement constraint I is imposed at the designated joint (JT(I,1)).

XJT(I,3) = Global Z component of the unit vector which is in the direction for which the displacement constraint I is imposed at the designated joint (JT(I,1)).

XJT(I,4) = 0.0

These components are used to transform the global displacement components at a constrained joint to a displacement component in the displacement constraint direction at this joint which will then be compared to the actual constraint value.

For a rotational constraint (IDTYP(I)=0, from Input Step 6), the following applies.

One data card where,

XJT(I,1) = Global X coordinate of joint JT(I,1).

XJT(I,2) = Global Z coordinate of joint JT(I,1).

XJT(I,3) = Global X coordinate of joint JT(I,2).

XJT(I,4) = Global Z coordinate of joint JT(I,2).

Because the two joints lie on the same floor and floors are assumed to be horizontal, the vertical Y coordinates are assumed to be the same for the two joints JT(I,1) and JT(I,2). These coordinates are used to compute the initial direction (relative to the positive X-axis) of a chord connecting the two joints so that when the deformed position is determined, the final direction of this chord may be found and the difference between the initial and final directions of the chord is the floor rotation (floors are taken as rigid bodies for these calculations), and is checked against the floor rotation constraint.

6. READ DC(I),IDTYP(I),JFN

FORMAT (F10.2,2I5)

One data card where,

DC(I) = Actual displacement constraint value for displacement constraint I. The units are inches for a translational constraint and radians for a rotational constraint.

IDTYP(I) = Displacement constraint type for displacement constraint I.

= 0 = Rotational constraint.

= 1 = Translational constraint.

JFN = Floor number where displacement constraint I exists. (This is only necessary when the displacement constraint is rotational. For a translational constraint, JFN=0.)

- -

Repeat steps 3 to 6 for each displacement constraint, a total of NDC times.

- -

7. READ J,COLID1(J),COLID2(J),AREA,RIX,RIY,RIZ

FORMAT (8X,I7,T1,2A4,7X,4F10.2)

One data card for each section in the column section table, a total of ICOL cards where,

J = Section number in column section table.

COLID1(J) = First four alphanumeric characters in column name.

COLID2(J) = Last four alphanumeric characters in column name.

AREA = Cross-sectional area of section J (in.^{2}).

RIX = Torsion constant of section J (in.^{4}).

RIY = Moment of inertia about minor axis of section J (in.^{4})

RIZ = Moment of inertia about major axis of section J (in.^{4}).

For an example of the contents of a column-section table, see Table B.1.

8. READ J,BMID1(J),BMID2(J),AREA,RIX,RIY,RIZ

FORMAT (8X,I7,T1,2A4,7X,4F10.2)

One data card for each section in the beam section table, a total of IBEAM cards where,

J = Section number in beam section table.

BMID1(J) = First four alphanumeric characters in beam name.

BMID2(J) = Last four alphanumeric characters in beam name.

AREA = Cross-sectional area of section J (in.^{2}).

RIX = Torsion constraint of section J (in.^{4}).

RIY = Moment of inertia about minor axis of section J (in.^{4}).

RIZ = Moment of inertia about major axis of section J (in.^{4}).

For an example of the contents of a beam section table, see Table B.2.

9. READ J,BRID1(J),BRID2(J),AREA,RIX,RIY,RIZ

FORMAT (8X,I7,2A4,7X,4F10.2)

One data card for each section in the bracing section table, a total of IBR cards where,

J = Section number in bracing section table.

BRID1(J) = First four alphanumeric characters in brace name.

BRID2(J) = Last four alphanumeric characters in brace name.

AREA = Cross-sectional area of section J (in.^{2}).

RIX = Torsion constant of section J (in.^{4}).

RIY = Moment of inertia about minor axis of section J (in.^{4}).

RIZ = Moment of inertia about major axis of section J (in.^{4}).

For examples of the contents of a bracing section table, see Tables B.3 and B.4.

- -

Steps 10 to 17 are necessary to input structure data, including number of joints, number of members, member properties, released directions, etc., as well as all loading conditions. The input formats shown in these steps are from the modified Weaver(15) programs for analysis which are used as an example of a user supplied analysis program and were used for all example problems discussed in Chapter 5.

- -

10. READ SN TS

FORMAT (2I3)

One data card where,

SN = Structure number. (Because the design system may design only one structure during any single execution of the program, this will <u>always</u> be equal to <u>1</u>.)

TS = Type of structure. (Structure type here refers to plane frame, space truss, space frame, etc., as defined by the terminology of Weaver(15). However, because the kinematic condensation is limited to space frames only, this will <u>always</u> be <u>6</u>, which indicates a space frame structure type.)

11. READ M,NJ,NR,NRJ,E,G

FORMAT (4I3,2F7.0)

One data card where,

M = Total number of members in the structure to be designed.

NJ = Total number of joints in the structure to be designed.

NR = Number of reactions. Sum total of restrained global displacement components from each joint in the structure.

NRJ = Number of reaction joints. Sum total of all joints in the structure with at least one restrained global displacement component. (A space frame joint has 6 possible global displacement components. If at a joint, any one of these 6 is not free to displace, that joint is included in the sum of NRJ joints.)

E = Modulus of elasticity. (This input implies that all members in the structure have the same modulus of elasticity.) (kips/inches^2)

G = Shear modulus of elasticity. (This input implies that all members in the structure have the same shear modulus of elasticity.) (kips/inches^2)

12. READ J X(J),Y(J),Z(J)

FORMAT (I3,3F7.0)

One data card for each joint in the structure, a total of NJ cards where,

J = Joint number.

X(J) = X-coordinate of joint J.

Y(J) = Y-coordinate of joint J.

Z(J) = Z-coordinate of joint J.

- -

For each member in the frame (a total of M), only Input Step 13 is specified if AA = 0 from Input Step 13; and if AA = 1 from Input Step 13, then Input Steps 13 and 14 must be specified.

- -

13. READ I,JJ(I),JK(I),AA,ITAB(I),ISEC(I),RHO,U

FORMAT (6I5,2F10.2)

One data card where,

I = Member number.

JJ(I) = Negative incident joint number of member I.

JK(I) = Positive incident joint number of member I.

AA = Beta angle indicator. (See Gere and Weaver(6) for beta angle discussion)

= 0 = Beta angle is zero.

= 1 = Beta angle is nonzero.

ITAB(I) = Number of the section table which contains the member properties for member I. (This number must be either a 16, which indicates the column section table, a 17, which indicates the beam section table, or an 18, which indicates the bracing section table. All three tables were input previously during Input Steps 7, 8 and 9.)

ISEC(I) = Number of the section within the section table which specifically identifies the name and member properties of member I. (These properties are Area, I_x, I_y and I_z.)

RHO = Mass density factor for member I in pounds per cubic ft. (e.g. 490.0 for 490 lbs./ft.^{3}).

U = Unit cost factor for member I in dollars per pound (e.g. 0.20 for \$0.20/lb.).

- -

If AA = 0, go to Input Step 13 for next member. If AA = 1, go to Input Step 14 next.

- -

14. READ I,XP,YP,ZP

FORMAT (I3,3F7.0)

One data card where,

I = Member number (Same as from Input Step 13).

XP = X-coordinate of a point that lies in one of the principal planes of the member I but is not on the centroidal axis of the member itself. (See Gere and Weaver (6), pp. 291 - 293).

YP = Y-coordinate of a point that lies in one of the principal planes of the member I but is not on the centroidal axis of the member itself.

ZP = Z-coordinate of a point that lies in one of the principal planes of the member I but is not on the centroidal axis of the member itself.

- -

- -

Repeat Input Steps 13, or 13 and 14, for each member in the structure, a total of M times.

- -

15. READ K,RL(6*K-5),RL(6*K-4),RL(6*K-3),RL(6*K-2),RL(6*K-1),RL(6*K)

FORMAT (7I3)

One data card for each joint with at least one restrained global displacement component, a total of NRJ cards where,

K = Joint number of a joint with <u>at least one</u> restrained global displacement component.

RL(6*K-5) = Joint restraint condition of global X displacement component at joint K.

= 0 = Unrestrained

= 1 = Restrained

RL(6*K-4) = Joint restraint condition of global Y displacement component at joint K.

= 0 = Unrestrained

= 1 = Restrained

RL(6*K-3) = Joint restraint condition of global Z displacement component at joint K.

= 0 = Unrestrained

= 1 = Restrained

RL(6*K-2) = Joint restraint condition of global X rotation component at joint K.

= 0 = Unrestrained

= 1 = Restrained

RL(6*K-1) = Joint restraint condition of global Y rotation component at joint K.

= 0 = Unrestrained

= 1 = Restrained

RL(6*K) = Joint restraint condition of global Z rotation component at joint K.

= 0 = Unrestrained

= 1 = Restrained

At least one of the values of RL(6*K-5) to RL(6*K) must be equal to 1.

- -

Steps 16 and 17 are repeated for each loading condition, a total of NLS (from Input Step 2) times.

- -

16. READ NLJ,NLM

FORMAT (2I3)

One data card where,

NLJ = Number of loaded joints. (A loaded joint is defined as any joint having an external force applied to it in one or more global displacement directions.)

NLM = Number of loaded members. (Because member loads are not permitted in the design system (see Chapter 2), this parameter must always be 0.)

17. READ K,A(6*K-5),A(6*K-4),A(6*K-3),A(6*K-2),A(6*K-1),
A(6*K)

FORMAT (I3,6F10.3)

One data card for each loaded joint, a total of NLJ cards where,

K = Joint number of a loaded joint.

A(6*K-5) = Applied force in global X direction at joint K.

A(6*K-4) = Applied force in global Y direction at joint K.

A(6*K-3) = Applied force in global Z direction at joint K.

A(6*K-2) = Applied moment in global X direction at joint K.

A(6*K-1) = Applied moment in global Y direction at joint K.

A(6*K) = Applied moment in global Z direction at joint K.

Applied external loading conditions will probably have many component loads at a single joint. However, virtual loads will usually have only a single component load at a joint. For a joint translational constraint, only one joint will be loaded. For a floor rotational constraint, two joints will be loaded to produce a couple.

- -

Repeat Steps 16 and 17 for each loading condition, a total of NLS times.

- -

If it is desired that any (1 or more) of the analyses is to be executed by the kinematic condensation procedure, then additional data is necessary. Otherwise input is complete.

- -

- -

So, if kinematic condensation is to be executed, and if the transformation matrix T is to be formed internally, go to step 22 (ITRANS = 1 or ITRANS = 2) next. Otherwise, go to input step 18.

- -

18. READ NJD,JD

    FORMAT (2I5)

    One data card where,

    NJD = Number of joints with <u>at least one</u> dependent displacement component.

    JD = Total number of dependent displacement components in the structure.

19. READ (IDEP(J,I),I = 1,8)

    FORMAT (8I5)

    One data card for each joint with at least one dependent displacement component, a total of NJD cards where,

    IDEP(J,1) = Joint number of a joint with at least one dependent displacement component.

    IDEP(J,2) = Total number of dependent displacement components at joint IDEP(J,1). ($1 \leq$ IDEP(J,2) ≤ 6).

    IDEP(J,3) = First local dependent displacement component direction at joint IDEP(J,1). ($1 \leq$ IDEP(J,3) ≤ 6).

    IDEP(J,4) = Second local dependent displacement component direction at joint IDEP(J,1). ($1 \leq$ IDEP(J,4) ≤ 6).

    IDEP(J,5) = Third local dependent displacement component direction at joint IDEP(J,4). ($1 \leq$ IDEP(J,5) ≤ 6).

IDEP(J,6) = Fourth local dependent displacement component direction at joint IDEP(J,1). ($1 \leq$ IDEP(J,6) ≤ 6).

IDEP(J,7) = Fifth local dependent displacement component direction at joint IDEP(J,1). ($1 \leq$ IDEP(J,7) ≤ 6).

IDEP(J,8) = Sixth local dependent displacement component direction at joint IDEP(J,1). ($1 \leq$ IDEP(J,8) ≤ 6).

The local dependent displacement components are restricted to be between 1 and 6, inclusive, because they are specified as local components as in Fig. 3.2(b). In a case where IDEP(J,2) is less than six, all those entries in the IDEP(J,1) array which have no meaning are input as 0. For example, if IDEP(J,2) is four, only four of the possible six local components are dependent. Therefore, IDEP(J,7) and IDEP(J,8) are both input as 0. It is also necessary that the local dependent components (1 to 6) be input in ascending order (i.e, 1 3 5 6, rather than 3 1 5 6).

20. READ IRR,IFUL1,NFUL1,IP,NP

FORMAT (5I5)

One data card where,

IRR = Number of absolute columns in the transformation matrix $\underset{\sim}{T}$.

IFUL1 = Number of absolute rows in final partitioned row of the transformation matrix $\underset{\sim}{T}$.

NFUL1 = Number of absolute columns in final partitioned column of the transformation matrix $\underset{\sim}{T}$.

IP = Number of partitioned rows of the transformation matrix $\underset{\sim}{T}$.

NP = Number of partitioned columns of the transformation matrix $\underset{\sim}{T}$.

For a discussion of the partitioning of the transformation matrix $\underset{\sim}{T}$, see Section 3.3.

21. From Section 3.3, it is seen that the transformation matrix $\underset{\sim}{T}$ has row and column partitioning producing records that are numbered consecutively across each partitioned row with a maximum of six columns and rows, respectively, in any single record. Records are input in ascending numerical order and depending on the position and size of the record, the program chooses the correct one of the following formats, a,b,c or d to read in each individual record. The array T(I,J) used in the following four READ statements is an array which temporarily stores the record of the transformation matrix being read in, before that record is stored on disk. The individual records of the $\underset{\sim}{T}$ matrix are read in by <u>column</u>.

For a record of maximum size (6x6),

```
21a.READ        ((T(I,J),I = 1,6),J = 1,6)
    FORMAT      (6F10.3)
```

The number of data cards for this record is equal to the number of elements in the record (36 in this case), divided by 6 (from the format, there are 6 numbers per card), which is 6 data cards.

For the last record in any partitional row, which has six absolute rows, except the last partioned row,

```
21b.READ        ((T(I,J),I = 1,6),J = 1,NFUL1)
    FORMAT      (6F10.3)
```

The number of data cards for a record of this type is equal to the number of elements in the record, divided by 6 (from the format, there are 6 numbers per card). This will <u>always</u> result in an integer number of data cards.

For any record in the final partitioned row, which has six absolute columns, except for the last record,

21c.READ ((T(I,J),I = 1,IFUL1),J = 1,6)

FORMAT (6F10.3)

The number of data cards for a record of this type is equal to the number of elements in the record, divided by 6(from the format, there are 6 numbers per card). This will <u>always</u> result in an integer number of data cards.

For the final record,

21d.READ ((T(I,J),I = 1,IFUL1),J = 1,NFUL1)

FORMAT (6F10.3)

The number of data cards for a record of this type is equal to the number of elements in the record divided by 6 (from the format, there are 6 numbers per card). This may not result in an integer number of data cards. Should this occur, the number of data cards is equal to the next highest integer number.

For a kinematic condensation analysis with the T matrix input by the user (ITRANS = 0 from Input Step 1), input is complete.

---

22. READ NF,XP,ZP

FORMAT (I5,2F10,2)

One data card where,

NF = Number of floors in the structure which are being simulated as rigid (complete or in-plane rigidity) diaphragms. (NF$\leq$24).

XP = The X-coordinate of the arbitrary vertical Y-axis about which all rigid (complete or in-plane rigidity) floors are assumed to rotate.

ZP = The Z-coordinate of the arbitrary vertical Y-axis about which all rigid (complete or in-plane rigidity) floors are assumed to rotate.

Note that XP and ZP are the same variable names as shows in Input Step 14. Because of their temporary nature, no conflict occurs.

- -

Steps 23 and 24 are executed for each floor in the structure which is being simulated as a rigid (complete or in-plane rigidity) diaphragm, a total of NF times.

- -

23. READ IFN,JF(IFN),NCF

FORMAT (3I5)

One data card where,

IFN = Floor number.

JF(IFN) = Number of joints on floor IFN which are included in the simulation of a rigid (in-plane or completely rigid) diaphragm.

NCF = Number of data cards containing the JF(IFN) joints

(The format of inputting these joints is shown in the following step, Input Step 24) (NCF $\geq$ 1).

24. READ (JTN(J1),J1 = 1,16)

FORMAT (16I5)

These are NCF data cards where,

JTN = An array which temporarily contains as many as 16 entries (see format) which indicate joints on floor IFN. The joints may be input as single entities, or an ascending sequentially numbered series of joints may be indicated by placing the first joint in one format slot and the last joint in the series in the next format slot, but preceded by a negative sign. This series type of input must all be on a single card and may not carry over to the next card (i.e., The first joint in the series may not be on one card, with the last joint in the series preceeded by a negative sign on the following card). Also, if more than one card is specified (NCF$\geq$1) all cards except the last card must contain 16 entries.

- -

Repeat steps 23 and 24 for each floor in the structure, which is being simulated as a rigid (complete or in-plane rigidity) diaphragm, a total of NF times.

- -

Input is now complete.

- -

APPENDIX B

SECTION PROPERTIES

The following tables (B.1, B.2, B.3, and B.4) contain the column, beam, and bracing section tables used in the examples presented in Chapter 5.

Note that,

I_x = torsion constant

I_y = minor principal axis moment of inertia

I_z = major principal axis moment of inertia

All sections contained herein are standard AISC sections. Column and beam section names are the same as the AISC designation. However, section names used for bracing sections are changed for convenience and these names along with their AISC designation are listed in Tables B.3 and B.4. Also recognize that bracing sections in Tables B.3 and B.4 are double angle sections with long legs back to back. Note that I_x, I_y, and I_z for these bracing sections are all 0.01. This is necessary to simulate a pinned-pinned condition for braces because of the limited capability, relative to member end releases, of the user provided stiffness analysis programs included in the system as an example (Chap. 4).

Table B.1. Economy Column Sections

Section Name	Weight (lb./ft.)	Area (in.^{2})	I_x (in.^{4})	I_y (in.^{4})	I_z (in.^{4})
6WF20	20	5.90	0.24	13.30	41.70
8WF24	24	7.06	0.34	18.20	82.50
8WF28	28	8.23	0.53	21.60	97.80
8WF31	31	9.12	0.53	37.00	109.70
8WF35	35	10.30	0.77	42.50	126.50
10WF39	39	11.48	0.97	44.90	209.70
12WF40	40	11.77	0.96	44.10	310.10
14WF43	43	12.65	1.05	45.10	429.10
14WF48	48	14.11	1.44	51.30	484.90
14WF53	53	15.59	1.93	57.50	542.10
12WF58	58	17.06	2.10	107.40	476.10
14WF61	61	17.94	2.19	107.30	641.50
14WF74	74	21.76	3.86	133.50	796.80
14WF78	78	22.94	3.52	206.90	851.20
12WF79	79	23.22	3.85	216.40	663.00
14WF84	84	24.71	4.41	225.50	928.40
12WF99	99	29.09	7.45	278.20	858.50
14WF111	111	32.65	7.48	454.90	1266.50
14WF119	119	34.99	9.20	491.80	1373.10
14WF127	127	37.33	11.10	527.60	1476.70
14WF136	136	39.98	13.50	567.70	1593.00
14WF142	142	41.85	14.20	660.10	1672.20
14WF150	150	44.08	16.70	702.50	1786.90
14WF158	158	46.47	19.50	745.00	1900.60
14WF167	167	49.09	22.80	790.20	2020.80
14WF176	176	51.73	26.50	837.90	2149.60
14WF184	184	54.07	30.30	882.70	2274.80
14WF193	193	56.73	34.70	930.10	2402.40
14WF202	202	59.39	39.60	979.70	2538.80
14WF211	211	62.07	44.80	1028.60	2671.40
14WF219	219	64.36	49.90	1073.20	2798.20
14WF228	228	67.06	56.20	1124.80	2942.40
14WF237	237	69.69	62.60	1174.80	3080.90
14WF246	246	72.33	69.70	1226.60	3228.90
14WF264	264	77.63	85.30	1331.20	3526.00
14WF287	287	84.37	108.00	1466.50	3912.10
14WF314	314	92.30	140.00	1631.40	4399.40
14WF320	320	94.12	137.00	1635.10	4141.70
14WF342	342	100.59	178.00	1806.90	4911.50
14WF370	370	108.78	222.00	1986.00	5454.20
14WF398	398	116.98	272.00	2169.70	6013.70

Table B.1. Continued

Section Name	Weight (lb./ft.)	Area (in.^{2})	I_x (in.^{4})	I_y (in.^{4})	I_z (in.^{4})
14WF426	426	125.25	330.00	2359.50	6610.30
14WF455	455	133.73	396.00	2561.20	7214.90
14WF500	500	146.95	514.00	2882.70	8234.10
14WF550	550	161.75	670.00	3256.70	9443.10
14WF605	605	177.85	869.00	3680.90	10842.30
14WF665	665	195.51	1120.00	4166.20	12477.70
14WF730	730	214.65	1450.00	4716.80	14371.40

Table B.2. Economy Beam Sections

Section Name	Weight (lb./ft.)	Area (in.^{2})	I_x (in.^{4})	I_y (in.^{4})	I_z (in.^{4})
6JR4.4	4.4	1.30	0.01	0.17	7.30
8JR6.5	6.5	1.92	0.02	0.34	18.70
10JR9	9	2.64	0.03	0.61	39.00
12JR11.8	11.8	3.45	0.04	0.98	72.00
10B15	15	4.40	0.10	2.79	68.80
12B16.5	16.5	4.86	0.11	2.79	105.30
14B17.2	17.2	5.05	0.11	2.65	147.30
14B22	22	6.47	0.21	6.40	197.40
16B26	26	7.65	0.26	8.71	298.10
14WF30	30	8.81	0.38	17.50	289.60
16B31	31	9.12	0.46	11.57	372.50
14WF34	34	10.00	0.57	21.30	339.20
16WF36	36	10.59	0.55	22.10	446.30
16WF40	40	11.77	0.79	26.50	515.50
18WF45	45	13.24	0.89	31.90	704.50
18WF50	50	14.71	1.25	37.20	800.60
21WF55	55	16.18	1.24	44.00	1140.70
21WF62	62	18.23	1.83	53.10	1326.80
24WF68	68	20.00	1.86	63.80	1814.50
24WF76	76	22.37	2.70	76.50	2096.40
27WF84	84	24.71	2.79	95.70	2824.80
27WF94	94	27.65	4.06	115.10	3266.70
30WF99	99	29.11	3.78	116.90	3988.60
30WF108	108	31.77	5.02	135.10	4461.00
30WF116	116	34.13	6.43	153.20	4919.10
33WF118	118	34.71	5.32	170.30	5886.90
33WF130	130	38.26	7.37	201.40	6699.00
36WF135	135	39.70	7.03	207.10	7796.10
36WF150	150	44.16	10.10	250.40	9012.10
36WF160	160	47.09	12.40	275.40	9738.80
36WF170	170	49.98	15.10	300.60	10470.00
36WF182	182	53.54	18.40	327.70	11281.50
36WF194	194	57.11	22.30	355.40	12103.40
36WF230	230	67.73	28.60	870.90	14988.40
36WF245	245	72.03	34.70	944.70	16092.20
36WF260	260	76.56	41.60	1020.60	17233.80
36WF280	280	82.32	52.60	1127.50	18819.30
36WF300	300	88.17	64.20	1225.20	20290.20

Table B.3. Unequal Leg Double Angle Bracing Sections

AISC Designation	Section Name	Weight (lb./ft.)	Area (in.^{2})	I_x, I_y, I_z (in.^{4})
L3x2x3/16	3UAN6.1	6.1	1.80	.01
L3x2½x¼	3UAN9.0	9	2.62	.01
L4x3x¼	4UAN11.6	11.6	3.38	.01
L3x2½x3/8	3UAN13.2	13.2	3.84	.01
L4x3x5/16	4UAN14.4	14.4	4.18	.01
L4x3½x5/16	4UAN15.4	15.4	4.50	.01
L4x3x3/8	4UAN17.0	17.0	4.96	.01
L4x3½x3/8	4UAN18.2	18.2	5.34	.01
L4x3x7/16	4UAN19.6	19.6	5.74	.01
L4x3½x7/16	4UAN21.2	21.2	6.18	.01
L4x3x½	4UAN22.2	22.2	6.50	.01
L6x3½x3/8	6UAN23.4	23.4	6.84	.01
L6x4x3/8	6UAN24.6	24.6	7.22	.01
L5x3x½	5UAN25.6	25.6	7.50	.01
L5x3½x½	5UAN27.2	27.2	8.00	.01
L6x4x7/16	6UAN28.6	28.6	8.36	.01
L7x4x7/16	7UAN31.6	31.6	9.24	.01
L8x4x7/16	8UAN24.4	34.4	10.12	.01
L8x4x½	8UAN39.2	39.2	11.50	.01
L8x4x3/4	8UAN57.4	57.4	16.88	.01

Table B.4. Equal Leg Double Angle Bracing Sections

AISC Designation	Section Name	Weight (lb./ft.)	Area (in.^{2})	I_x, I_y, I_z (in.^{4})
L2x2x¼	2AN6	6.38	1.88	.01
L2x2x5/16	2AN7	7.84	2.30	.01
L2½x2½x5/16	2.5AN10	10.00	2.94	.01
L3x3x3/8	3AN14	14.4	4.22	.01
L3½x3½x3/8	3.5AN17	17.0	4.96	.01
L4x4x3/8	4AN19	19.6	5.72	.01
L3½x3½x½	3.5AN22	22.2	6.50	.01
L4x4x½	4AN25	25.6	7.50	.01
L5x5x7/16	5AN28	28.6	8.36	.01
L5x5x½	5AN32	32.4	9.50	.01
L6x6x7/16	6AN34	34.4	10.12	.01
L6x6x½	6AN39	39.2	11.50	.01
L8x8x½	8AN52	52.8	15.50	.01
L6x6x3/4	6AN57	57.4	16.88	.01
L8x8x5/8	8AN65	65.4	19.22	.01
L6x6x1	6AN74	74.8	22.00	.01
L8x8x7/8	8AN90	90.0	26.46	.01
L8x8x1	8AN102	102.0	30.00	.01
L8x8x9/8	8AN113	113.8	33.46	.01

APPENDIX C

DISK FILES

Many disk file operations are performed during the execution of the design system. Large quantities of information are stored out of core to conserve in core memory and to permit large scale structures to be designed. References to the information stored on the disk files were made in Chapters 2, 3 and 4.

The length of each of the 17 files described herein are found in the define file statements in the listing (program MAIN) and the sizes shown are consistent with the maximum problem size detailed in Appendix A (input) and Appendix F (program listing and comments).

In addition to the file description, the subprogram name and line number within that subprogram which references each disk file (READ or (WRITE) is also presented.

File 11

Number of Records: Total number of degrees of freedom, n(i.e., (6*NJ)-NR).

Maximum Size of Record: Total number of degrees of freedom, n.

Purpose: Individual rows of the stiffness matrix are stored on records on file 11 by subprogram STIFF in packed form, if a kinematic condensation analysis is required. These rows are then read in, unpacked (i.e., spread out and each row filled in) and again written to file 11 by subprogram UNPACK. Finally, these rows are read in by STFDES to begin the formation of the kinematically reduced stiffness matrix.

Points of Access and Line Numbers:

1. STIFF 0539

   WRITE (11'IREK)(S(IREK,J), J = 1,UBW)

2. UNPACK 0015

   READ (11'IREC)(S(I,J),J = 1,UBW)

3. UNPACK 0051

   WRITE (11'I)(STEMP(IJ,J),J = 1,K)

4. STFDES 0048

   READ (11'KM)(STEMP(IK,J),J = 1,K)

File 12

Number of Records: Number of records on and to the right of the diagonal record in the unpacked original structure stiffness matrix. See Fig. 3.7 for an example.

Maximum Size of Record: 36 words.

Purpose: When a partitioned row of the packed stiffness matrix is read in core and unpacked, a void exists to the left of the diagonal record. Appropriate partitioning of the columns in each partitioned row of this unpacked matrix produces records to the right of the diagonal record, and these are stored on file 12. Those stored records are later read in as needed to complete a subsequent unpacked partitioned row. Refer to Section 3.5 and Figs. 3.8 to 3.11 for a more detailed discussion and illustrations.

Points of Access and Line Numbers:

1. RD1AG 0016

```
WRITE (12'IREC)((STEMP(I,J),I = 1,IR(II)),J = LL,LSUM)
```

2. LD1AG 0025

```
READ (12'IREC)((DUM(K,L),K = 1,IR(JJ)),L = 1,IR(II))
```

File 13

Number of Records: Number of partitioned rows (IP) x Number of partitioned columns (NP) of the $\underset{\sim}{T}$ matrix.

Maximum Size of Record: 36 words

Purpose: File 13 stores the transformation matrix $\underset{\sim}{T}$ in record form discussed in Section 3.3 and shown in Fig. 3.6. This matrix is necessary in forming the kinematically reduced stiffness equations, Eq. 3.8.

Points of Access and Line Numbers:

1. FORMT 0091

   WRITE (13'JREC)((S(I,J),I = 1,IA),J = KK,LL)

2. FORMT 0106

   WRITE (13'JREC)((T(I,J),I = 1,6),J = 1,6)

3. FORMT 0110

   WRITE (13'JREC)((T(I,J),I = 1,6),J = 1,NFUL1)

4. FORMT 0116

   WRITE (13'JREC)((T(I,J),I = 1,IFUL1),J = 1,6)

5. FORMT 0120

   WRITE (13'JREC)((T(I,J),I = 1,IFUL1),J = 1,NFUL1)

6. RPKKT 0035

   READ (13'ITREC)((T(IK,IJ),IK = KK,LL),IJ = 1,NA)

7. TTRKT 0028

   READ (13'ITREC)((T(IK,IJ),IK = KK,LL),IJ = 1,NA)

8. RLOADV 0021

   READ (13'ITREC)((T(IK,IJ),IK = KK,LL),IJ = 1,NA)

File 14

Number of Records: Number of partitioned rows x Number of partitioned columns of the product $K_{22}T$.

Maximum Size of Record: 36 words.

Purpose: One step in the process of forming the kinematically reduced stiffness matrix is to post multiply portions of the reordered, unpacked stiffness matrix by T. That portion of the stiffness matrix below the horizontal partition separating independent and dependent equations (below K_{11} and K_{12} and above K_{21} and K_{22} in Eq. 3.2) is operated on by T and formed into records of K_{22} T. These records are stored into file 14 for later access and pre-multiplication by T^t to complete the matrix triple product $T^tK_{22}T$ shown in Eq. 3.8. See Section 3.4 and Fig. 3.6 for details.

Points of Access and Line Numbers:

1. STORE 0026

   ```
   WRITE (14'KTREC)((S(K,J),K = 1,IR(II)),J = KK,LL)
   ```

2. TTRKT 0048

   ```
   READ (14'KTREC)((S(K,J),K = 1,IR(II)),J = KK,LL)
   ```

File 15

Number of Records: Number of loading conditions x 2.

Maximum Size of Record: Number of displacement degrees of freedom ((6*NJ)-NR).

Purpose: File 15 stores all load vectors. Each individual load vector is stored on one record (record LN) in its original form for a stiffness analysis and its corresponding kinematically reduced form is stored on another record (record LN+NLS) for use in kinematic condensation analysis. For details, see Section 3.4.

Points of Access and Line Numbers:

1. LDATA 0141

   WRITE (15'LN)(AC(I),I = 1,N)

2. RLOADV 0006

   READ (15'LN)(S(I,K),K = 1,N)

3. RLOADV 0047

   WRITE (15'LN1)(S(3,K),K = 1,L)

4. MODFR2 0036

   READ (15'LN1)(AC(I),I = 1,NROW)

5. MODFR2 0048

   READ (15'LN)(AC(I),I = 1,N)

File 16

Number of Records: One record for each section in column section table.

Maximum Size of Record: 4 words.

Purpose: File 16 stores the area, I_x, I_y, and I_z for each section in the user input column section table (see Table B.1 for an example). The record number is identical to the section number in the table.

Points of Access and Line Numbers:

1. PTABLE 0015

   WRITE (16'J) AREA,RIX,RIY,RIZ

2. SDATA 0157

   READ (ITABI'ISECI) AX(I),IX(I),IY(I),IZ(I)

3. VIRTWK 0009

   READ (ITABI'ISECI) AREA,RIX,RIY,RIZ

4. VIRTWK 0038

   READ (ITABI'NSEC) AREAN,RIXN,RIYN,RIZN

5. COSTWT 0020

   READ (16'J) AREA,RIX,RIY,RIZ

File 17

Number of Records: One record for each section in beam section table.

Maximum Size of Record: 4 words.

Purpose: File 17 stores the area, I_x, I_y, and I_z for each section in the user input beam section table (see Table B.2 for an example). The record number is identical to the section number in the table.

Points of Access and Line Numbers:

1. PTABLE 0025

   WRITE (17'J) AREA,RIX,RIY,RIZ

2. SDATA 0157

   READ (ITABI'ISECI) AX(I),IX(I),IY(I),IZ(I)

3. VIRTWK 0009

   READ (ITABI'ISECI) AREA,RIX,RIY,RIZ

4. VIRTWK 0038

   READ (ITABI'NSEC), AREAN,RIXN,RIYN,RIZN

5. COSTWT 0025

   READ (17'J) AREA,RIX,RIY,RIZ

File 18

Number of Records: One record for each section in bracing section table.

Maximum Size of Record: 4 words.

Purpose: File 18 stores the area, I_x, I_y, and I_z for each section in the user input bracing section table (see Tables B.3 and B.4 for examples). The record number is identical to the section number in the table.

Points of Access and Line Numbers:

1. PTABLE 0036

   WRITE (18'J) AREA,RIX,RIY,RIZ

2. SDATA 0157

   READ (ITABI'ISECI) AX(I),IX(I),IY(I),IZ(I)

3. VIRTWK 0009

   READ (ITABI'ISECI) AREA,RIX,RIY,RIZ

4. VIRTWK 0038

   READ (ITABI'NSEC) AREAN,RIXN,RIYN,RIZN

5. COSTWT 0015

   READ (18'J) AREA,RIX,RIY,RIZ

File 19

Number of records: One record for each structural member.

Maximum Size of Record: 3 words.

Purpose: Each member has a density factor and a cost factor. These values in addition to member length are stored on file 19 on the record number which is equal to the member number that is described with these factors.

Points of Access and Line Numbers:

1. SDATA 0171

   WRITE (19'I) RHO,U,XL

2. VIRTWK 0012

   READ (19'I) RHO,U,XL

3. COSTWT 0012

   READ (19'I) RHO,U,XL

File 20

Number of Records: One record for each joint in the structure.

Maximum Size of Record: 3 words.

Purpose: Joint coordinates (Global X,Y, and Z for each joint) are stored on file 20 in the record number which is equal to the joint number they locate. These coordinates are necessary to compute all non-zero elements in the $\underset{\sim}{T}$ matrix when that matrix is formed internally by the program.

Points of Access and Line Numbers:

1. SDATA 0082

   ```
   WRITE (20'J) X(J),Y(J),Z(J)
   ```

2. FORMT 0057

   ```
   READ (20'MJK1) XI,YI,ZI
   ```

Files 21-27

Number of Records: On each file, there is one record for each structural member.

Maximum Size of Record: 5 words.

Purpose: Each file from 21 to 27 corresponds to a single loading condition. A maximum of seven loading conditions are permitted. Among these seven loading conditions, a maximum of three may be virtual loading conditions, each of which corresponds to a displacement constraint. Associated with each virtual loading condition may be a maximum of four applied external loading conditions (see Chapter 2 for discussion of Q-P loading table). Member end forces which include axial force, y and z axis bending moments at the negative incident end, and y and z axis bending moments at the positive incident end are stored for later access during subsequent virtual work calculations(see Chapter 2). The record number is equal to the member number for which the member end forces are applicable.

Points of Access and Line Numbers:

1. RESULT 0280

   WRITE (JFILE'I) AM(1),AM(5),AM(6),AM(11),AM(12)

2. VIRTWK 0011

   READ (IFILE'I) MEAU1,MEAU2,MEAU3,MEAU5,MEAU6

3. VIRTWK 0016

   READ (JFILE'I) MEA1,MEA2,MEA3,MEA5,MEA6

APPENDIX D

DOCUMENTATION OF SUBPROGRAMS

This appendix contains descriptions of all subprograms used in the stiffness design system. They are divided up into their functional areas which are DESIGN, KINEMATIC CONDENSATION ANALYSIS, and USER PROVIDED STIFFNESS ANALYSIS (Fig. 4.2).

D.1 DESIGN Subprograms

This section contains all subprograms which perform overall control, and specific design functions. They are described in the order,

1. MAIN
2. PTABLE
3. TSCHK
4. DSENCO
5. NDCHG
6. DVWCHG
7. INCMEM
8. DEXACT
9. VIRTWK
10. COSTWT
11. OUTPUT

Name: MAIN

Author: Barry Adelson

Date: September 1976

Language: IBM Fortran

Description and Logic: This is the overall controlling program for all phases of the design system which are briefly described here. There are 17 different files defined to store pertinent data and avoid the impossibility of having all information in core at once. All design parameters are input and immediately printed at the outset of the program. A data check is made to verify that all quantities are within allotted limits. Error messages, detailed below, result if any value exceeds its bounds. Column, beam, and bracing section tables are input in a program especially designed to handle that data. As soon as all counters are initialized to zero, the design process begins with an analysis either by kinematic condensation analysis, or by stiffness analysis, depending on what is specified by the user. Immediately following analysis, displacements atconstraint locations are checked to see if they are less than or equal to constraint values and if so, the process ends. Otherwise, the design process begins with the gradient search cycle, as error terms are calculated to compensate somewhat for the approximate nature of the succeeding computations done by the virtual work procedures. Programs are called sequentially which evaluate displacement sensitivity coefficients, Q_i's, decide which constraint directions have member changes associated with them, and update deflections at all constraint locations due to those member increases. This set of programs is basically a tight loop of virtual work calculations. The loop is repeated until all constraints appear to be satisfied, thus ending a gradient search cycle. If permitted by the user, an analysis of either type (kinematic condensation or stiffness analysis) is completed to verify that a solution has been found or that displacements are still excessive. Should more changes be necessary, the gradient search cycle composed of repeated virtual work loops between analyses is executed again. When a final solution is reached, a program is called to output and summarize in tabular form, the results of the design process. Intermediate output from the main program itself allows the engineer to note in more detail how the design progresses.

Program Output: The program title is output first followed in tabular form by all design parameters which are as follows: Number of columns, beams, and braces in each respective section table, number of approximate kinematic condensation analyses per exact stiffness analysis, maximum number of total analyses permitted (kinematic condensation + exact stiffness analyses), transformation matrix parameter, maximum number of member updates permitted per analysis, member change parameter, initial and final analysis type parameters, number of loading conditions including unit loads, number of displacement constraints, unit load magnification factor , tolerance of displacement convergence in percent and initial error percent. Following this list, the location, magnitude, number and direction of displacement constraint is printed after input, plus the loading number of the unit load associated with each translational constraint. For rotational constraints, substituted for direction are joint coordinates for two joints indicating the chord which is used to measure rotation. Immediately after an analysis is completed, a summary is printed which includes the displacements at each constraint location resulting from its associated loads and compared with the limiting constraint values. Then after each virtual work loop, following the output from another program of the exact member numbers which are changed, another table contains the approximate displacements from virtual work at each constraint location, evolving from its associated loads, error terms used to modify those deflections, and constraint values themselves. At the completion of a gradient search cycle, a summary of member changes by member type as well as analysis number and type are output. Once cost and weight data are printed from another program, a displacement summary showing the computed deflections by analysis at the beginning of the cycle in addition to the approximate deflections at the end of the cycle for each associated applied load for each constraint are compared with corresponding constraint values.

Program Length: 291 Source statements + comments

Calls from Program: PTABLE, MODFR2 (analysis), DSENCO, NDCHG, DVWCHG, COSTWT, OUTPUT

Calls to Program: None

Error Procedures: If the following occurs:

Initial or final analysis type parameter is not equal to 0 or 1, member change parameter is not equal to 0 or 1, number of loadings is less than 1 or greater than 7, number of displacement constraints is less than

1 or greater than 3, Transformation matrix parameter is not equal to 0 or 1 or 2, the vector describing the direction of a translational constraint is not of unit length, the following message results.

ALL INPUT DATA ARE NOT WITHIN ALLOWED LIMITS...CHECK.

Name: PTABLE

Author: Barry Adelson

Date: September 1976

Language: IBM Fortran

Descrption and Logic: This program sets up consecutive loops which control the input of column, beam and bracing section tables in that order, containing area, and moments of inertia about the local x, y, and z axes. These tables are stored on disk files.

Program Output: Listing of column, beam and bracing section tables.

Program Length: 39 Source statements+comments.

Calls from Program: None

Calls to Program: MAIN

Error Procedures: None

Name: TSCHK

Author: Barry Adelson

Date: September 1976

Language: IBM Fortran

Description and Logic: This is essentially a checking program which makes sure that the table and section specified for each member in the structure are within allowable limits as dictated by design input parameters.

Program Output: None

Program Length: 31 Source statements + comments

Calls from Program: None

Calls to Program: MODFR2 (analysis program)

Error Procedures: If member table or section is incorrectly specified for any member in the structure the following message is printed.

IMPROPER MEMBER TABLE OR SECTION SPECIFIED---

MEMBER #____, TABLE #____, SECTION #____

This continues until all members are checked. An error here ultimately causes the program to cease. Should a member be at its maximum size, all of its displacement sensitivity coefficients are set equal to +10000 to prevent that member from being selected to be increased.

Name: DSENCO

Author: Barry Adelson

Date: September 1976

Language: IBM Fortran

Description and Logic: This program organizes the calculation of displacement sensitivity coefficients Q_i's immediately after an analysis (kinematic condensation or stiffness analysis) has been performed for use in the upcoming gradient search cycle. One Q_i is computed by virtual work means, for each member, for each associated load of each displacement constraint, except under 2 conditions. No Q_i's are computed for any members already at their maximum allowable size and none are computed which are related to a displacement constraint that already has been met. Once all appropriate Q_i's have been calculated, the most negative one associated with each individual displacement constraint and the member that produces it, are stored for later use, as it is these members which are to be changed. Later on in the program, as members are increased, new Q_i's are calculated in another program. On subsequent passes through DSENCO, after a virtual work loop, the most negative of the existing Q_i's are checked and again the most negative for each constraint is ascertained. Note that only immediately after an analysis, does this program control the calculation of all Q_i's for use in the gradient search and no matter where a Q_i is called for, the actual calculation is performed in subprogram VIRTWK.

Program Output: None

Program Length: 25 Source statements + comments

Calls from Program: VIRTWK

Calls to Program: MAIN

Error Procedures: None

Name: NDCHG

Author: Barry Adelson

Date: September 1976

Language: IBM Fortran

Description and Logic: This program sorts out information and decides which displacement constraints will have member changes associated with them during the upcoming virtual work cycle. Generally, if the input parameter MEMCHG is equal to 1, then one member change will occur for each displacement constraint and if MEMCHG is equal to 0, the member with overall the most negative Q_i will be the only one increased, no matter which constraint it is associated with and this data is stored in array ICHG. Special conditions cause a variation in the basic procedure. Should MEMCHG equal to 1 and a single member have the most negative dsenco for 2 or more constraints, that member is increased only once during the subsequent virtual work loop because it will affect both constraints by a significant amount anyway. Also, if a constraint is already satisfied, as ascertained by either virtual work or exact calculations, no member change will correspond to that constraint.

Program Output: None

Program Length: 31 Source statements + comments

Calls from Program: None

Calls to Program: MAIN

Error Procedures: None

Name: DVWCHG

Author: Barry Adelson

Date: September 1976

Language: IBM Fortran

Description and Logic: This program dictates when a member change is to take place, calculates changes and updates deflections due to the member increases, manages computation of Q_i's for the newly increased members and keeps tract of the total number of members changed since the last analysis. As each member is increased, displacements at all constraint locations are affected with a resulting decrease. Deflections from each associated load of each constraint are updated, whether or not a member change is associated with that constraint. A loop is made over all constraints.

Program Output: None

Program Length: 39 Source statements + comments

Calls from Program: VIRTWK, INCMEM

Calls to Program: MAIN

Error Procedures: None

Name: INCMEM

Author: Barry Adelson

Date: September 1976

Language: IBM Fortran

Description and Logic: The actual increase by one in size in section in the appropriate section table, column, beam or brace, occurs in this program. Should that member reach its maximum size upon the increase,all of its Q_i's are set equal to +10000. A cumulative tabulation of member changes by member type is carried out in this program.

Program Output: As a member is increased, its number, table and section are printed.

Program Length: 41 Source statements + comments.

Calls from Program: None

Calls to Program: DVWCHG

Error Procedures: Should it be determined that a member has a table specified as anything other than 16, 17, or 18, column, beam or brace respectively, before execution ceases, the following error message is printed.

IMPROPER MEMBER TABLE OR SECTION SPECIFIED ---

MEMBER #____, TABLE #____, SECTION #____.

If it happens that the member chosen to be increased is at its maximum size already, this means that all members are at their maximum size and thus the following message results.

ALL MEMBERS ARE AT MAXIMUM SIZE.

Name: DEXACT

Author: Barry Adelson

Date: September 1976

Language: IBM Fortran

Description and Logic: The purpose of this program is to compute the exact value of translation or rotation, whichever is specified, at constraint locations for each associated loading condition of each individual constraint. The importance of this program is evident when the direction of a translational constraint is not parallel to either the X, Y, or Z global axes, and when the type of constraint is rotational.

Program Output: None

Program Length: 76 Source statements + comments

Calls from Program: None

Calls to Program: MODFR2 (analysis)

Error Procedures: None

Name: VIRTWK

Author: Barry Adelson

Date: September 1976

Language: IBM Fortran

Description and Logic: All virtual work calculations are performed in this program. When necessary, member end actions from a particular applied loading condition, those arising from a unit load relating to a displacement constraint, member properties defined by table and section numbers are read in from various disk files and mathematically combined, as in Eq. 2.17 to determine the contribution to displacement of the member in question at the designated displacement constraint location, eventually encompassing all constraints. Factors of density, cost and length plus member properties from the next largest section in the same table are called from other disk files to determine a Q_i for that member as shown in Eq. 2.25. Repeated calls to this program ultimately result in Q_i's for all members, for each associated load of each displacement constraint.

Program Output: None

Program Length: 50 Source statements + comments

Calls from Program: None

Calls to Program: DSENCO, DVWCHG

Error Procedures: None

Name: COSTWT

Author: Barry Adelson

Date: September 1976

Language: IBM Fortran

Description and Logic: Structure weight and cost are evaluated in this program. Factors of density, cost and length, plus member properties, as designated by table and section number, are read into core from disk file storage. Weight is computed as the product of density, area and length, and this quantity is simply multiplied by the unit cost factor to arrive at the cost of a single member. A loop is set up to include all members with branches to separate and accumulate weight and cost by member type, finally summing each category to reach total weight and cost for the entire frame under consideration. This program is called at the outset of the design and again following each gradient search cycle.

Program Output: Subtotals of weight and cost for columns, beams and braces, and a final sum total for each are printed.

Program Length: 44 Source statements + comments

Calls fromProgram: None

Calls to Program: MAIN, MODFR2 (analysis), OUTPUT

Error Procedures: None

Name: OUTPUT

Author: Barry Adelson

Date: September 1976

Language: IBM Fortran

Description and Logic: This is a program which controls output of data vital to the design process. The program is executed twice during the complete design process. Initial and final member properties are output from this program as a loop is completed over all members. Depending on member type, column, beam or brace, and section number, member properties of area and 3 moments of inertia are read into core from disk storage. Another file contains density, unit cost and length on a record pertaining to each member. All these quantities are output in tabular form. On the second pass through the program, number and type of analysis plus a breakdown of member changes by column, beam and bracing section, for the recently completed final design, are also printed, and immediately following,a call is made to another program to evaluate final structure cost and weight in addition to member properties.

Program Output: See previous discussion.

Program Length: 66 Source statements + comments

Calls from Program : COSTWT

Calls to Program : MODFR2 (analysis) MAIN

Error Procedures: None

D.2 KINEMATIC CONDENSATION ANALYSIS Subprograms

This section contains all subprograms which perform kinematic condensation analysis. They are described in the order,

1. STFDES
2. CONCHK
3. GLOBAL
4. ORDER
5. UNPACK
6. LDIAG
7. RDIAG
8. DIAG
9. ROWBND
10. RPKKT
11. STORE
12. TTRKT
13. RLOADV
14. KFN
15. EXPRD
16. FORMT

Name: STFDES

Author: Barry Adelson

Date: September 1976

Language: IBM Fortran

Description and Logic: This program directs all activities pertaining to the kinematic reduction process. The first steps include calling programs to perform consistency checks on input parameters, convert local dependent displacement directions to global directions, and assemble a permutation vector, all in preparation for and preceding the formation of the transformation matrix in another program. Should an error occur during the data check, execution ceases. These calls are skipped on subsequent executions of the kinematic condensation. Next, an array is formed to identify diagonal record numbers of the stiffness matrix, which are derived from its row and column partitioning scheme. Because the stiffness matrix is stored in packed form on disk by the user supplied analysis program (See Chapter 4), these are necessary as a basis for numbering other records in the matrix, used when a separate program unpacks and fills in the void to the left of the diagonal elements, for each individual partitioned row. A sufficient number of single rows of the stiffness matrix, saved on disk by the unpacking program, are read into core one at a time, in their new order determined by the permutation vector, until one partitioned row is complete. It is then ascertained whether the row in question is above, below, or straddling the imaginary horizontal boundary between independent and dependent portions of the stiffness matrix. Once this position is known, a series of calls are made to programs that assess the band width of the partitioned row after its columns are reordered, perform the appropriate post multiplications by the transformation matrix and transfer those absolute rows of the independent segment of the reduced matrix to their final locations, and store on disk the multiplication results in the dependent sector of the condensed matrix. When the loop encompassing all partitional rows is concluded, a final program is called to finish the matrix triple product, producing the final reduced stiffness matrix in core.

Program Output: None

Program Length: 70 Source statements + comments

Calls from Program: CONCHK, GLOBAL, ORDER, FORMT, UNPACK, ROWBND, RPKKT, STORE, TTRKT

Calls to Program: MODFR2 (analysis)

Error Procedures: None

Name: CONCHK

Author: Barry Adelson

Date: September 1976

Language: IBM Fortran

Description and Logic: This program performs a consistency check on all data input in conjunction with the kinematic condensation procedure. Those values which are not within specified ranges produce error messages detailed below. The program is split into two major sections, one concerning the internal formulation of the transformation matrix, the other deals with parameters necessary for user input of such a matrix.

Program Output: See error procedures below.

Program Length: 248 Source statements + comments

Calls from Program: None

Calls to Program: STFDES

Error Procedures: The first set of error messages are concerned with parameters necessary for internal formulation of the transformation matrix. If the number of stories in the frame is given as a negative number, the following appears.

NO. OF FLOORS = ________, WHICH IS IMPOSSIBLE

If the total number of joints specified as being on a particular floor or rigid body does not equal the sum of all those listed on data cards, the message below appears.

NO. OF JOINTS ON CARDS DOES NOT EQUAL NUMBER READ IN. INFORMATION ON DATA CARD (Values from the data card are printed out in the input format.)

If the first value on a data card specifying joints on a floor or rigid body is equal to 0, the error is noted as follows.

DATA INPUT ERROR FOR FLOOR NO. ___ ,

FIRST JOINT ON FLOOR SPECIFIED AS ZERO

INFORMATION ON DATA CARD (Values from the data card are printed out using input format.)

If in the middle of a list of joints on a data card, a 0 appears, the message below follows.

DATA INPUT ERROR FOR FLOOR NO. ___ ,

ZERO SPECIFIED AS A JOINT

INFORMATION ON DATA CARD (Values from the data card are printed out using input format.)

If the first value on a data card specifying joints on a floor or rigid body is negative, the error message below is the result.

DATA INPUT ERROR FOR FLOOR NO. ___ ,

FIRST JOINT SPECIFIED AS NEGATIVE

INFORMATION ON DATA CARD (Values from the data card are printed out using input format.)

It is possible to signify a series of consecutive joints by indicating the first joint in the series in one format slot, and the final joint in the series in the following format slot on the same card, with a negative sign in front of it. For instance 5,-8 would indicate joints 5, 6, 7 and 8. However, if the absolute value of the second number specified is less than that of the first, the following message is printed.

DATA INPUT ERROR FOR FLOOR NO. ___ ,

CONSECUTIVE JOINTS NOT IN ORDER

INFORMATION ON DATA CARD (Values from the data card are printed out using input format.)

Should two consecutive entries be negative, another message is printed.

DATA INPUT ERROR FOR FLOOR NO. ___ ,

TWO CONSECUTIVE JOINTS SPECIFIED AS NEGATIVE

INFORMATION ON DATA CARD (Values from the data card are printed out using input format.)

Any of the above errors causes termination of the program after all floors are checked. Otherwise execution continues by internally forming an array which is directly related to the desired simulated behavior of the structure and precedes the ultimate internal assemblage of the transformation matrix for either the completely rigid floor or planar rigidity cases. A final check is made in this portion of the program to see that no joint is specified twice. If so, the following is printed.

INPUT ERROR -- JOINT ___ APPEARS ___ TIMES.

Any joint appearing multiple times causes an error exit to occur, and this error message results from a duplicate joint which appears in either major portion of the program.

The following set of error indicators are related to quantities used when a transformation matrix is externally input by the user. The number of joints with dependent displacements, NJD, must be greater than 0 and the number of dependent displacements must be greater than or equal to NJD, if such is not the case, the following message results.

NJD OR JD HAS BEEN INCORRECTLY SPECIFIED

NJD = ___ JD = ___

Each joint, which has at least one unrestrained dependent displacement component, is input on a single card, with those components in ascending order. The next several errors are related to a violation of this format. The joints themselves need not be in any special order. If the number of dependent displacements is inadvertently specified as zero at any of these joints, the error message below is printed.

NEGATIVE OR ZERO NO. OF DEPENDENT DISPLACEMENTS SPECIFIED

IDEP(J,2) = ___

Local components must be designated as a number between one and six, inclusive, or the following error indicator is printed.

LOCAL DEPENDENT DISPLACEMENT COMPONENT IMPROPERLY SPECIFIED AS A NO. LESS THAN 1 OR GREATER THAN 6

JOINT INFORMATION = ___ ___ ___ ___ ___ ___ ___ ___

Should these local components not be in ascending order, the user receives the following message.

DEPENDENT DISPLACEMENT COMPONENTS NOT IN ASCENDING ORDER

JOINT INFORMATION = ___ ___ ___ ___ ___ ___ ___ ___

If a local component chosen to be dependent is found to be restrained, the user is informed as follows.

A DEPENDENT DISPLACEMENT COMPONENT HAS BEEN SPECIFIED AS RESTRAINED

JOINT INFORMATION = ___ ___ ___ ___ ___ ___ ___ ___

A check is made to insure that the sum total of all dependent displacements is equal to the number which was read in and if not the following message results.

NO. OF DEPENDENT DISPLACEMENT COMPONENTS ADDED UP FROM EACH OF NJD JOINTS IS NOT EQUAL TO JD READ IN. NJD = ___ JD = ___

Name: GLOBAL

Author: Barry Adelson

Date: September 1976

Language: IBM Fortran

Description and Logic: This program converts local dependent displacement component numbers to global unknown components from 1 up to the number of degrees of freedom. The global direction is dependent upon the joint at which the component is located and the cumulative number of restraints previous to the direction under consideration. Arrays are formed which store the number of released directions at each joint, and are directly related to the number of absolute rows and columns in each partitioned row and column. The total number of partitioned rows and columns in the original stiffness matrix is determined here.

Program Output: None.

Program Length: 73 Source statements + comments.

Calls from Program: None.

Calls to Program: STFDES

Error Procedures: None.

Name: ORDER

Author: Barry Adelson

Date: September 1976

Language: IBM Fortran

Description and Logic: Using the global directions of the dependent displacement components, an array is formed to dictate the reordering scheme necessary for the stiffness matrix, before pre and post multiplication by the transformation matrix is executed as a part of the kinematic condensation procedure.

Program Output: None.

Program Length: 28 Source statements + comments.

Calls from Program: None.

Calls to Program: STFDES

Error Procedures: None.

Name: UNPACK

Author: Barry Adelson

Date: September 1976

Language: IBM Fortran

Description and Logic: This program reads in from the appropriate disk file enough absolute rows of the original stiffness matrix, stored in packed form by the user supplied analysis program, in order to generate one partitioned row. Details of partitioning are found in Chapter 3. Operations carried out in this program are on the packed form of the original stiffness matrix. In effect the partitioned row is expanded such that elements are in their actual column positions, leaving a void to the left of the diagonal elements. Two major arrays are used, one to store the packed partitioned row, and one to store the unpacked arrangement. Other programs are called which transpose the diagonal record and fill in the remaining void further to the left of this position, taking advantage of symmetry of the stiffness matrix by using transposed records to the right of the diagonal from previous partitioned rows. The records referred to evolve from the column partitioning which is identical to row partitioning. Once the entire partitioned row is filled, each individual row is written out to the same disk file from which it came for later modification. A loop is set up to manipulate all partitioned rows as described above.

Program Output: None.

Program Length: 55 Source statements + comments.

Calls from Program: LDIAG, DIAG, RDIAG

Calls to Program: STFDES

Error Procedures: None.

Name: LDIAG

Author: Barry Adelson

Date: September 1976

Language: IBM Fortran

Description and Logic: This program calculates the proper record numbers necessary to fill the void created to the left of the diagonal after a partitioned row is unpacked. Required nonzero records are read in from disk, transposed and inserted in the proper position of the partitioned row under consideration until that row is filled.

Program Output: None.

Program Length: 40 Source statements + comments.

Calls from Program: None.

Calls to Program: UNPACK

Error Procedure: None.

Name: RDIAG

Author: Barry Adelson

Date: September 1976

Language: IBM Fortran

Description and Logic: This program organizes the unpacked partitioned row under consideration into records based on column partitioning. Those nonzero records to the right of the diagonal one are written to a disk file for later use in filling the void to the left of the diagonal in subsequent partitioned rows.

Program Output: None.

Program Length: 18 Source statements + comments.

Calls from Program: None.

Calls to Program: UNPACK

Error Procedures: None.

Name: DIAG

Author: Barry Adelson

Date: September 1976

Language: IBM Fortran

Description and Logic: This program transposes the diagonal record of an unpacked partitioned row as a contribution to filling in the void to the left of the diagonal elements created when that row is unpacked.

Program Output: None.

Program Length: 27 Source statements + comments.

Calls from Program: None.

Calls to Program: UNPACK

Error Procedures: None.

Name: ROWBND

Author: Barry Adelson

Date: September 1976

Language: IBM Fortran

Description and Logic: The purpose of this program is to assess the band width of each individual partitioned row of the rearranged stiffness matrix before it is pre or post multiplied by the transformation matrix. By knowing the band width, which is defined as the largest number of columns between the diagonal and the last nonzero element, inclusive, of any single row in the partitioned row in question, many multiplications by zero may be omitted.

Program Output: None.

Program Length: 36 Source statements + comments.

Calls from Program: None.

Calls to Program: STFDES

Error Procedures: If it is determined that the reordered matrix has a number less than or equal to zero on the diagonal, an error message is printed, REORDERED MATRIX HAS A ZERO ON THE DIAGONAL. This causes the program to cease any further execution.

Name: RPKKT

Author: Barry Adelson

Date: September 1976

Language: IBM Fortran

Description and Logic: This program begins the actual process of kinematic condensation once all preliminaries are completed. Upon entering the program, a partitioned row of the original stiffness matrix is in core with rows reordered. If the row is completely above the horizontal partition separating the independent and dependent equations, it is physically column reordered and the $\underset{\sim}{K}_{11}$ segment is stored in a temporary array with $\underset{\sim}{K}_{12}\underset{\sim}{T}$ multiplication following, producing elements to complete the partitioned row of the reduced matrix, which is stored in packed form in a temporary array in core. The physical reordering is completed only for those elements contained in $\underset{\sim}{K}_{11}$. If the row is entirely below the horizontal partition, all elements of $\underset{\sim}{K}_{22}\underset{\sim}{T}$ are obtained by matrix multiplication and temporarily preserved in core, later on in another program to be stored on disk, as they are needed ultimately for completion of the triple product $\underset{\sim}{T}^t\underset{\sim}{K}_{22}\underset{\sim}{T}$. Due to symmetry of the stiffness matrix, $\underset{\sim}{T}^t\underset{\sim}{K}_{21}$ is not necessary. For all multiplications in this program, a pseudo column reordering, facilitated by the permutation vector previously assembled, picks the proper elements from the rows of the original stiffness matrix as they occur in the rearranged stiffness matrix, to multiply by the transformation matrix. The band width of the partitioned row in question, ascertained prior to this point, is advantageously used to minimize the number of calculations. Partitioned columns of $\underset{\sim}{T}$ are cycled into core one at a time for these computations. For any row which straddles the horizontal partition, each section is treated as a full partitioned row according to where it is located.

Program Output: None.

Program Length: 66 Source statements + comments

Calls from Program: None.

Calls to Program: STFDES

Error Procedures: None.

Name: STORE

Author: Barry Adelson

Date: September 1976

Language: IBM Fortran

Description and Logic: A partitioned row of the stiffness matrix, once it reaches this program, is reordered and appropriate columns have been post multiplied by the transformation matrix. Should the partitioned row under investigation be completely above the horizontal partition, it is transferred to its exact position in the array storing the final reduced stiffness matrix. If the row is entirely below the imaginary partition between independent and dependent equations, this implies that it is composed of elements evolving from $\underset{\sim}{K}_{22}\underset{\sim}{T}$ multiplication. Hence it is organized into records which are sent out to a disk storage file for succeeding premultiplication by the transpose of the transformation matrix, leading to the eventual completion of the $\underset{\sim}{T}^t\underset{\sim}{K}_{22}\underset{\sim}{T}$ triple product. In the case of a row straddling the partition, each individual segment, above and below, is treated as a full partitioned row according to where it is located. There are as many partitioned rows of $\underset{\sim}{K}_{22}\underset{\sim}{T}$ as there are full partitioned rows of the stiffness matrix below the horizontal partition, except that one additional partitioned row exists in the situation of a row straddling that line. In all cases, there are the same number of partitioned columns of $\underset{\sim}{K}_{22}\underset{\sim}{T}$ as there are of the transformation matrix. These characteristics govern the formation of disk record structure, as records are numbered across each partitioned row.

Program Output: None.

Program Length: 46 Source statements + comments

Calls from Program: None.

Calls to Program: STFDES

Error Procedures: None.

Name: TTRKT

Author: Barry Adelson

Date: September 1976

Language: IBM Fortran

Description and Logic: This program has as its function to complete the $\underset{\sim}{T}^t \underset{\sim}{K}_{22} \underset{\sim}{T}$ triple product and store the result in packed form in the final condensed matrix array. The $\underset{\sim}{K}_{11}$ and $\underset{\sim}{K}_{12}\underset{\sim}{T}$ portion of that matrix is previously completed in the proper form upon entering this program. An outer loop increments on partitioned columns of the transformation matrix, which can also be defined as partitioned rows of the transpose of that matrix, while an inner loop increments over partitioned columns of the $\underset{\sim}{K}_{22}\underset{\sim}{T}$ matrix, each of which is stored on a separate disk file and has an individual array set aside for one of its column or row partitions for use in core. To save space, as new partitions are called, they simply replace the existing ones. The record numbers composing these designated partitions for each matrix are calculated internally by the program. Ultimately, when the triple product is accomplished, the reduced matrix in packed form exists in a special array in preparation for use in solving for independent displacements and displacement measures. Because it is in packed form, care is taken to calculate only those elements which fall on the diagonal or above. Also the band width of the final matrix is ascertained as an aid in solving.

Program Output: None.

Program Length: 85 Source statements + comments

<u>Calls from Program</u>: None.

<u>Calls to Program</u>: STFDES

<u>Error Procedures</u>: None.

Name: RLOADV

Author: Barry Adelson

Date: September 1976

Language: IBM Fortran

Description and Logic: The load vector, as well as the stiffness matrix, must be modified to correctly apply the kinematic condensation procedure. One load vector at a time is read in from the proper disk file where it was stored by the user supplied analysis program. This vector is physically reordered, as dictated by the permutation vector described earlier. The elements associated with the dependent displacements are premultiplied by partitioned columns of the transformation matrix, which can also be referred to as partitioned rows of that matrix, as they are read in one at a time for the necessary multiplications. Each partition replaces the previous one until all are processed. The resulting reduced loading vector is then sent out to the same disk file, with a different record number.

Program Output: None.

Program Length: 49 Source statements + comments

Calls from Program: None.

Calls to Program: MODFR2 (analysis)

Error Procedures: None.

Name: KFN

Author: Barry Adelson

Date: September 1976

Language: IBM Fortran

Description and Logic: During the consistency data check, it is necessary repeatedly to determine the overall global direction of a local direction, that has been specified as a dependent displacement, for the purpose of determining whether or not it is restrained or unrestrained. This FUNCTION subprogram is used for that purpose.

Program Output: None

Program Length: 8 Source statements + comments

Calls from Program: None

Calls to Program: CONCHK

Error Procedures: None

Name: EXPRD

Author: Barry Adelson

Date: September 1976

Language: IBM Fortran

Description and Logic: When the condensed stiffness matrix and condensed load vector are solved, the resulting vector contains values for the independent joint displacements and the displacement measures. For the displacement measures to be converted into dependent joint displacements, they must be premultiplied by the transformation matrix. This is accomplished by employing partitioned rows of that matrix, one at a time, each partition replacing the previous one, until the final product is formulated. At this point, all displacements are calculated, but are not in the proper sequence, so a physical reordering process, using the permutation vector as a guide, puts the displacements back into their original order. The final order is critical because of the way in which displacements are used to compute member end forces in an analysis program.

Program Output: None.

Program Length: 49 Source statements + comments

Calls from Program: None.

Calls to Program: MODFR2 (analysis)

Error Procedures: None.

Name: FORMT

Author: Barry Adelson

Date: September 1976

Language: IBM Fortran

Description and Logic: This program contains two major segments, one to manage input related to the internal assemblage of the transformation matrix for 2 specialized conditions, and another to control input of parameters plus the desired transformation matrix itself. In either case, the matrix has partitioned rows and columns arranged such that they contain 6 absolute rows and columns respectively, except for the last one of each which may contain less because of overall dimensions. Records, based on this partitioning scheme and numbered consecutively across each row, are written to a disk file for later use in forming the reduced stiffness matrix and reduced load vectors, as well as modifying displacement measures, creating dependent displacements.

Program Output: None.

Program Length: 123 Source statements + comments

Calls from Program: None.

Calls to Program: STFDES

Error Procedures: None.

D.3 USER PROVIDED STIFFNESS ANALYSIS Subprograms

This Section contains all subprograms which perform exact stiffness analysis and which represent an example of the types of programs a user should provide and interface with the other two parts of the design system as outlined in Chapter 4 and Sections D.1 and D.2. These subprograms are described in the order,

1. MODFR2
2. SDATA
3. STIFF
4. DCBAND
5. LDATA
6. RESULT
7. SBAND

Name: MODFR2

Author: William Weaver(15) (modified by Barry Adelson)

Date: September 1976 (modified)

Language: IBM Fortran

Description and Logic: This is a modified user provided program which has the function of controlling all procedures necessary to perform an analysis of a given structure for joint displacements and member forces. It was modified to control not only an exact stiffness analysis, but also direct a complete kinematic condensation analysis. The modifications are described in detail in Chapter 4.

Program Output: Output from this program consists of headings for the following information; displacements and member end actions, initial structure weight and cost data, and loading conditions.

Program Length: 62 Source statements + comments

Calls from Program: RLOADV, SBAND, EXPRD, RESULT, DEXACT, SDATA, TSCHK, OUTPUT, COSTWT, STIFF, LDATA, STFDES, DCBAND

Calls to Program: MAIN

Error Procedures: None

Name: SDATA

Author: William Weaver (15)(modified by Barry Adelson)

Date: September 1976 (modified)

Language: IBM Fortran

Description and Logic: This is a user provided program which has the function of controlling input and output of all structure data including the following: number of members, number of joints, number of reactions, number of reaction joints, modulus of elasticity, shear modulus, joint coordinates, member incidences, beta angle, member table and section size (i.e. member properties), unit cost and density factors for each member, and a list of restrained directions. Joint coordinates are written to a disk file for later use in forming the transformation matrix if a kinematic condensation is to be performed. The length of each member is calculated, and that along with its cost and density factors are written to another disk file for subsequent use in virtual work calculations. Details of modifications made to the program are found in Chapter 4.

Program Output: See discussion above.

Program Length: 267 Source statements + comments

Calls from Program: None.

Calls to Program: MODFR2

Error Procedures: None.

Name: STIFF

Author: William Weaver(15)(modified by Barry Adelson)

Date: September 1976 (modified)

Language: IBM Fortran

Description and Logic: This is a user provided program which has the function of forming the global structure stiffness matrix in packed form and determining its band width. If a kinematic condensation is to be performed, the stiffness matrix in packed form taking advantage of band width is written out to a disk file, one row per record. Details of the modifications to this program are found in Chapter 4.

Program Output: None.

Program Length: 542 Source statements + comments

Calls from Program: None.

Calls to Program: MODFR2

Error Procedures: None.

Name: DCBAND

Author: William Weaver (15)

Date: September 1976

Language: IBM Fortran

Description and Logic: This is a user provided program which performs a Cholesky decomposition on either the original stiffness matrix or kinematically condensed stiffness matrix in packed form, whichever is required. There were no modifications to this program.

Program Output: None.

Program Length: 24 Source statements + comments

Calls from Program: None.

Calls to Program: MODFR2

Error Procedures: If it is found that an element on the diagonal of the decomposed stiffness matrix is equal to or less than zero, termination of the program ensues with the following message,

DCBAND FAILS

Name: LDATA

Author: William Weaver (15) (modified by Barry Adelson)

Date: September 1976 (modified)

Language: IBM Fortran

Description and Logic: This is a user supplied program which has the function of controlling input of loads (joint loads only for reasons explained in Chapter 2), and formulation and output of load vectors. Load vectors are written to a disk file, one vector to a record, for subsequent kinematic condensation if that is required by the user. Details of the modifications to this program are found in Chapter 4.

Program Output: See discussion above.

Program Length: 143 Source statements + comments

Calls from Program: None.

Calls to Program: MODFR2

Error Procedures: None.

Name: RESULT

Author: William Weaver (15) (modified by Barry Adelson)

Date: September 1976 (modified)

Language: IBM Fortran

Description and Logic: This is a user supplied program which uses the global joint displacements to calculate both member end forces and reactions. Appropriate member end forces for each member are written to a disk file, one record for each set of member forces, for later use in the design system as described in Chapter 2. All joint displacements, member end forces and reactions are output from this program. Modifications are found in Chapter 4.

Program Output: See discussion above.

Program Length: 354 Source statements + comments

Calls from Program: None.

Calls to Program: MODFR2

Error Procedures: None.

<u>Name</u>: SBAND

<u>Author</u>: William Weaver (15)

<u>Date</u>: September 1976

<u>Language</u>: IBM Fortran

<u>Description and Logic</u>: This is a user supplied program which properly decomposes the load vector (original or kinematically reduced) and uses the banded decomposed stiffness matrix (original or kinematically reduced) to solve for displacements. In the case of a stiffness analysis, actual joint displacements result. For a kinematic condensation analysis, independent joint displacements and displacement measures result. There are no modifications to this program.

<u>Program Output</u>: None.

<u>Program Length</u>: 31 Source statements + comments

<u>Calls from Program</u>: None.

<u>Calls to Program</u>: MODFR2

<u>Error Procedures</u>: None.

APPENDIX E

DEFINITION OF COMMON VARIABLES

All COMMON variables used by the design system are shown in the subprogram listings in Appendix F, and are of generally two types. One type (Type 1) are those variables that were originally developed for the user supplied stiffness analysis subprograms, and the other type (Type 2) are variables that were developed specifically for the overall stiffness design system. Many Type 1 variables are not used by the design system programs and consequently are not described herein, while other Type 1 variables are used and are described. In the description of COMMON variables which follows, all Type 1 variables are marked with an asterisk (*).

Now, as noted in Appendix F, all subprograms of the design system may be associated with three categories, namely DESIGN, KINEMATIC CONDENSATION ANALYSIS, and USER PROVIDED STIFFNESS ANALYSIS PROGRAMS (also see Fig. 4.2). For convenience and clarity, all COMMON variables used primarily by the DESIGN subprograms, both Type 1 and Type 2, are described in Section E.1 in the same relative order that they appear in the program listings, while all COMMON variables used primarily by the KINEMATIC CONDENSATION subprograms, both Type 1 and Type 2, are described in Section E.2. Only those COMMON variables in the USER SUPPLIED STIFFNESS ANALYSIS subprograms which are used by the DESIGN and KINEMATIC CONDENSATION subprograms are described in Sections E.1 and E.2.

E.1 DESIGN Subprogram Variables in COMMON

Q(I,J) = The minimum displacement sensitivity coefficient for member I corresponding to deflection constraint location J, as chosen from a series of coefficients for each associated applied loading condition.

DELEX(I,J) = Exact displacement at deflection constraint location I, due to Jth associated load.

DERROR(I,J) = Error term for displacement constraint location I, resulting from Jth associated load.

DVW(I,J) = Virtual work displacement at displacement constraint location I, due to Jth associated load.

DELVWN(I,J) = New virtual work deflection contribution at displacement constraint location I, due to Jth associated load as contributed by the member which was most recently changed.

DELVW(I,J) = Old virtual work deflection contribution at displacement constraint location I, due to Jth associated load as contributed by the member which was most recently changed.

XJT(I,J) = This array takes on different values depending on whether a translational or rotational type of constraint is specified.
First assume a translational constraint:

XJT(I,1) = Global X component of the unit vector which describes the direction for which the displacement constraint I is imposed at the constraint joint JT(I,1).

XJT(I,2) = Global Y component of the unit vector which describes the direction for which the displacement constraint I is imposed at the constraint joint JT(I,1).

XJT(I,3) = Global Z component of the unit vector which describes the direction for which the displacement constraint I is imposed at the constraint joint JT(I,1).

XJT(I,4) = 0.0

For a rotational constraint:

XJT(I,1) = Global X coordinate of joint JT(I,1), where I is the displacement constraint number.

XJT(I,2) = Global Z coordinate of joint JT(I,1), where I is the displacement constraint number.

XJT(I,3) = Global X coordinate of joint JT(I,2), where I is the displacement constraint number.

XJT(I,4) = Global Z coordinate of joint JT(I,2), where I is the displacement constraint number.

DC(I) = Value of displacement constraint at location I.

QDMIN(I) = Value of the most negative displacement sensitivity coefficient corresponding to displacement constraint I.

THETA(I) = Array used to help describe floor rotation as described below:

THETA(1) = Initial counter-clockwise angle measured between the positive X-direction and the direction defined by the chord connecting two specified joints on a floor, JT(I,1) and JT(I,2), for rotational constraint I.

THETA(2) = Final counter-clockwise angle measured between the positive X-direction and the direction defined by the chord connecting two specified joints on a floor, JT(I,1) and JT(I,2), for rotational constraint I.

FACT = Unit load magnification factor.

IREG = A counter used to store the cumulative number of exact stiffness analyses performed.

JCHK = A counter used to store the cumulative number of stiffness and kinematic condensation analyses.

IQPT(I,J) = An array used to relate displacement constraint numbers and other parameters as shown below:

IQPT(I,1) = Unit load number associated with displacement constraint I.

IQPT(I,2) = Number of applied loads for which displacement constraint I must be met.

IQPT(I,3) = Loading number of first applied load for which displacement constraint I must be met.

IQPT(I,4) = Loading number of second applied load for which displacement constraint I must be met, if more than one exists, otherwise it equals 0.

IQPT(I,5) = Loading number of third applied load for which displacement constraint I must be met, if more than two exist, otherwise it equals 0.

IQPT(I,6) = Loading number of fourth applied load for which displacement constraint I must be met, if more than three exist, otherwise it equals 0.

JT(I,J) = An array which takes on different values dependent upon whether a translational or rotational constraint is specified.
First assume a translational constraint:

JT(I,1) = Joint at which displacement constraint I is imposed.

JT(I,2) = 0

For a rotational constraint:

JT(I,1) = One joint specified on a floor where a rotational constraint exists.

JT(I,2) = A second joint specified on a floor where a rotational constraint exists.

IDTYP(I) = An array whose value indicates the constraint type of constraint I. When it is 0, a rotational constraint is indicated, when it is 1, a translational constraint is designated.

IOK(I) = An array which indicates whether or not displacement constraint I is satisfied. When it is 0, the constraint is not satisfied, when it equals 1, it is satisfied.

IMEM(I) = An array which indicates when it is necessary to compute all displacement sensitivity coefficients. This means one for each member and for each associated load of each displacement constraint. When it is equal to 0, all coefficients are computed, when equal to 1, the computation is skipped.

MQD(I) = The member number which has the most negative displacement sensitivity coefficient for displacement constraint I.

ICHG(I) = An array which indicates whether or not a member change is to be associated with constraint I. When it is equal to 0, a member change is made which corresponds to constraint I, when equal to 1, no member change is made for that direction.

IDC(I) = An array which indicates whether or not a particular load is an associated load of displacement constraint I under consideration, thus making it necessary to compute an exact value of displacement at that constraint location.

ICOL	=	Number of columns in column section table.
IBEAM	=	Number of beams in beam section table.
IBR	=	Number of braces in bracing section table.
NDC	=	Number of displacement constraints.
ITER	=	A counter used to store the cumulative number of member changes occurring in the program.
IDERIV	=	A variable which indicates whether the ensuing virtual work calculations are to compute, for the displacement constraint under consideration, displacement sensitivity coefficients, or the contribution to deflection of a particular member. When it is equal to 1, coefficients are computed, when equal to 0, deflections are computed.
MEMCHG	=	A variable which dictates the pattern of member changes in the virtual work loop. When it is equal to 0, only that member with the most negative sensitivity coefficient overall is changed, when equal to 1, there is one member change permitted for each displacement constraint.
IX	=	When the deflection contribution of a member is required, it is the value of thisvariable which determines whether that contribution is to be calculated before or after the member change. When it is equal to 0, that member's contribution is to be calculated before, when equal to 1, the contribution is to be calculated after the member is changed.
JMEM	=	A counter used to keep track of the total number of member changes occurring during a single gradient search cycle.
LL	=	Certain loops from 1 to the number of displacement constraints, with this variable as the indexing variable, make calls to programs which use this parameter to make calculations or determine array subscripts.
L	=	This variable contains, in special cases, the member number which is selected to be changed.
IX16	=	Variable necessary for direct access file 16.
IX17	=	Variable necessary for direct access file 17.
IX18	=	Variable necessary for direct access file 18.

IX19 = Variable necessary for direct access file 19.

IX20 = Variable necessary for direct access file 20.

IX21 = Variable necessary for direct access file 21.

IX22 = Variable necessary for direct access file 22.

IX23 = Variable necessary for direct access file 23.

IX24 = Variable necessary for direct access file 24.

IX25 = Variable necessary for direct access file 25.

IX26 = Variable necessary for direct access file 26.

IX27 = Variable necessary for direct access file 27.

IMAX(I) = The value of this array determines whether or not member I is at its maximum size. When it is equal to 0, the member is not at its maximum size, when equal to 1, the member is at its maximum.

NC = Cumulative number of column changes within a single gradient search cycle.

NB = Cumulative number of beam changes within a single gradient search cycle.

NBR = Cumulative number of brace changes within a single gradient search cycle.

NCS = Total number of column changes in the completed design.

NBS = Total number of beam changes in the completed design.

NBRS = Total number of brace changes in the completed design.

COLID1(I) = First 4 alphanumeric characters in the name of section I in column section table.

COLID2(I) = Last 4 alphanumeric characters in the name of section I in column section table.

BMID1(I) = First 4 alphanumeric characters in the name of section I in beam section table.

BMID2(I) = Last 4 alphanumeric characters in the name of section I in beam section table.

BRID1(I)	=	First 4 alphanumeric characters in the name of section I in bracing section table.
BRID2(I)	=	Last 4 alphanumeric characters in the name of section I in bracing section table.
ITAB(I)	=	Table number from which the section number of member I is found.
ISEC(I)	=	Section number for member I.
IFIRST	=	A variable which indicates whether or not the first full analysis has been performed. When it is equal to 0, the first analysis has not been completed, when equal to 1, the initial analysis has been completed.
M*	=	Total number of members in the structure.
E*	=	Modulus of elasticity of structural material.
DJ(I)*	=	Array containing the displacement at overall global displacement direction I, which includes restrained directions (in natural order).
IX11	=	Variable necessary for direct access file 11.
IX12	=	Variable necessary for direct access file 12.
IX13	=	Variable necessary for direct access file 13.
IX14	=	Variable necessary for direct access file 14.
IX15	=	Variable necessary for direct access file 15.

NLS*	=	Total number of loading conditions, including unit loads at constraint points.
LN*	=	Loading number under consideration.
IFLAG	=	This is the initial analysis type parameter. When it is equal to 0, an exact analysis is called for, when equal to 1, a kinematic condensation is required.
IERR	=	A variable which determines whether or not an error has been determined within the design system. When it is equal to 0, no error exists, when equal to 1, an error exists which leads to termination of execution prematurely.

ITRANS = Transformation matrix parameter. When it is equal to 0, the transformation matrix is to be input; when equal to 1, the transformation matrix is to be formed internally for the case of completely rigid floors; when equal to 2, the transformation matrix is to be formed internally for the case of in-plane rigidity of floors.

ISTFD = A counter which keeps track of the number of kinematic condensations performed.

E.2 KINEMATIC CONDENSATION Subprogram Variables in COMMON

RL(I)* = An array which indicates whether or not global displacement direction I is released or restrained. When it is equal to 0, the direction is released, when equal to 1, the direction is restrained.

CRL(I)* = Cumulative number of restrained directions up to and including global direction I, as I is increased from 1 to the number of global displacement directions in the entire structure.

N* = Total number of degrees of displacement freedom.

UBW* = Band width of the original stiffness matrix which is defined as the largest number of columns between the diagonal element and the last nonzero element, inclusive, in any row in the matrix.

NJ* = Total number of joints in the structure.

NROW = The sum of the number of independent displacements and displacement measures, when the kinematic condensation technique is applied, which is also equal to the number of rows and columns in the reduced stiffness matrix.

NUBW = Band width of the reduced stiffness matrix after kinematic condensation is applied, which is equal to the largest number of columns between the diagonal element and last nonzero element, inclusive, in any row in the matrix.

ST(I,J) = Array which stores the final reduced stiffness matrix after kinematic condensation is applied, with I being the row number and J being the column number.

D(I)* = A vector containing displacements at each degree of freedom for a stiffness analysis, or containing independent displacements and displacement measures for a kinematic condensation analysis.

IDEP(I,J) = An array containing information relating a joint containing dependent displacement components, and the local directions of those components, as described below, with I ranging from 1 to the number of joints with dependent displacements, NJD.

IDEP(I,1) = Joint number of a joint with at least one dependent displacement.

IDEP(I,2) = Total number of dependent displacements at joint IDEP(I,1).

IDEP(I,3) = First local dependent displacement direction at joint IDEP(I,1).

IDEP(I,4) = Second local dependent displacement direction at joint IDEP(I,1) if more than 1 exists, otherwise it equals 0.

IDEP(I,5) = Third local dependent displacement direction at joint IDEP(I,1) if more than 2 exist, otherwise it equals 0.

IDEP(I,6) = Fourth local dependent displacement direction at joint IDEP(I,1) if more than 3 exist, otherwise it equals 0.

IDEP(I,7) = Fifth local dependent displacement direction at joint IDEP(I,1) if more than 4 exist, otherwise it equals 0.

IDEP(I,8) = Sixth local dependent displacement direction at joint IDEP(I,1) if more than 5 exist, otherwise it equals 0.

NJD = Number of joints with dependent displacements.

IR(I) = Number of released directions at a joint containing at least one unrestrained direction, with I ranging from 1 to the number of joints with at least one released direction, NRJT. Joints are considered in ascending order with fully restrained joints skipped. The values in this array represent the number of absolute rows in a partitioned row of the original stiffness matrix.

JROW(I,J) = An array, as defined below, which contains the number and directions of release at each joint in the structure, where I varies from 1 to the number of joints in the structure.

JROW(I,1) = Total number of released directions at joint I.

JROW(I,2) = First released displacement direction at joint I in global terms if it exists.

JROW(I,3) = Second released displacement direction at joint I in global terms if it exists.

JROW(I,4) = Third released displacement direction at joint I in global terms if it exists.

JROW(I,5) = Fourth released displacement direction at joint I in global terms if it exists.

JROW(I,6) = Fifth released displacement direction at joint I in global terms if it exists.

JROW(I,7) = Sixth released displacement direction at joint I in global terms if it exists.

NRJT = Total number of joints in the structure with at least one released direction and also equal to the number of partitioned rows in the original stiffness matrix.

IOR(I) = An array (the permutation vector) containing the new row and column position of original row and column position of degree of freedom I.

JD = Total number of dependent displacements in the full structure.

INDEP = Total number of independent displacements in the entire structure.

IDIAG(I) = Diagonal record number in partitioned row I of the original stiffness matrix.

LRR = Total number of displacement measures in the entire structure, also equal to the number of absolute columns in the $\underset{\sim}{T}$ transformation matrix.

INR = The actual column number in the stiffness matrix of a column in the absolute row of the partitioned row which is currently being unpacked.

LNC = Current column index, J, of an element in array STEMP(I,J), used in the unpacking process.

KTREC = The current record number of $\underset{\sim}{K}_{22}\underset{\sim}{T}$ currently being used.

IPART = Integer whose value indicates whether or not the partitioned row under consideration is completely below the horizontal partition between independent and dependent equations. When it is equal to 0, the row is not completely below the partition, when equal to 1, it is completely below.

IJK = Number of absolute rows below the horizontal partition separating independent and dependent equations, which are contained in a partitioned row that straddles this boundary, if such a row exists.

KLM = Partitioned row number of the first partitioned row completely below the horizontal partition separating independent and dependent equations.

ICROW	=	The cumulative number of absolute rows of the original stiffness matrix from the first absolute row of the stiffness matrix to the last absolute row of the partitioned row currently being unpacked.
II	=	Row number of the partitioned row of the stiffness matrix, which is currently being unpacked and processed.
JJ	=	Column number of the partitioned column, within partitioned row II, which is currently being processed.
S(I,J)	=	Array used to store a single partitioned row of the packed stiffness matrix, before unpacking occurs.
LLD	=	Total number of absolute columns to the left of the diagonal record in partitioned row II which is currently being unpacked.
STEMP(I,J)	=	Array used to store unpacked partitioned row of the stiffness matrix.
DUM(I,J)	=	Array used to temporarily store a single record of the unpacked, completed stiffness matrix.
KUBW	=	Band width of reordered partitioned row of the stiffness matrix under consideration.
NP	=	Number of partitioned columns of $\underset{\sim}{T}$.
NFUL1	=	Absolute number of columns in the last partitioned column of $\underset{\sim}{T}$.
IP	=	Number of partitioned rows of $\underset{\sim}{T}$.
T(I,J)	=	Array used to store a partitioned column or partitioned row of $\underset{\sim}{T}$.
D1(I)	=	Vector of displacements with independent displacements listed first followed by dependent displacements.
IFUL1	=	Absolute number of rows in the last partitioned row of $\underset{\sim}{T}$.
JLIM	=	Sum of the number of independent displacements plus the number of displacement measures in the structure which is being designed.
NF	=	Number of floors in the structure which are assumed rigid (i.e. complete or in-plane rigidity for internal formulation of $\underset{\sim}{T}$ matrix.).

XP = The X-coordinate of the arbitrary vertical axis about which each floor is assumed to rotate, if rigid body motion is assumed for that floor.

ZP = The Z-coordinate of the arbitrary vertical axis about which each floor is assumed to rotate, if rigid body motion is assumed for that floor.

JTN(I) = Temporary input storage variable for joints on a floor which are assumed to displace in a rigid body pattern.

MJ(I,J) = Array which stores joint and floor information based on one of the 2 types of rigid body assumptions, if any, being made as described below.

For I = 1, a fully rigid floor is assumed.

For I = 2, in-plane floor rigidity is assumed.

MJ(I,1) = Joint for which a $\underset{\sim}{T}$ submatrix is currently being generated.

MJ(I,2) = Floor on which joint MJ(I,1) is located.

JF(I) = Number of joints on floor I which displace in a rigid body pattern (i.e. complete or in-plane rigidity).

ISJ(I,J) = Joint number of the J^{th} joint on floor I which displaces in a rigid body pattern (i.e. complete or in-plane rigidity).

APPENDIX F

LISTING OF SUBROUTINES

The subprograms in the following listing (also see Fig. 4.2) fall into three categories and are named below:

1. DESIGN

MAIN	DSENCO	INCMEM	COSTWT
PTABLE	NDCHG	DEXACT	OUTPUT
TSCHK	DVWCHG	VIRTWK	

2. KINEMATIC CONDENSATION ANALYSIS

STFDES	UNPACK	ROWBND	RLOADV
CONCHK	LDIAG	RPKKT	KFN
GLOBAL	RDIAG	STORE	EXPRD
ORDER	DIAG	TTRKT	FORMT

3. USER PROVIDED STIFFNESS ANALYSIS

MODFR2	STIFF	LDATA	SBAND
SDATA	DCBAND	RESULT	

The following additional limitations on the size of the problem which may be executed by the program as currently developed are described below. Also see Appendix A, List of Input, for the other limitations.

1. Maximum number of joints is 100.
2. Maximum number of members is 300.
3. Maximum number of columns in column section table is 48.
4. Maximum number of beams in beam section table is 38.

5. Maximum number of braces in bracing section table is 20.

6. Maximum number of global displacement components is 600 (Includes reaction components).

7. Maximum number of unknown displacements in reduced stiffness matrix is 360 (INDEP + IRR).

8. Maximum number of partitional rows or columns of $\underset{\sim}{T}$ is 60 (IP or NP).

9. Maximum number of records of $\underset{\sim}{T}$ is 576.

10. Maximum number of records of unpacked stiffness matrix is 5100.

11. Maximum number of records of $\underset{\sim}{K}_{22}\underset{\sim}{T}$ is 576.

12. Maximum number of "floors" if kinematic condensation is used is 24 (a floor is defined as a unit which has displacement measures associated with joints contained in it).

Note that these limitations may be changed by simply increasing the size of all subscripted variables representing the system data structure.

```
*DECK MAIN                                                              MAI0000
C     PROGRAM MAIN
      COMMON Q(300,3),DELEX(3,6),DERROR(3,6),DVW(3,6),DELVWN(3,6),      MAI0001
     1DELVW(3,6),XJT(3,4),DC(3),QDMIN(3),THETA(2),FACT,IREG,JCHK,
     2IOPT(3,6),JT(3,2),IDTYP(3),IOK(3),IMEM(3),MOD(3),ICHG(3),IDC(3),
     3ICOL,IBEAM,IBR,NDC,ITER,IDERIV,MEMCHG,IN,JMEM,LL,I,IX16,IX17,IX18,
     4IX19,IX20,IX21,IX22,IX23,IX24,IX25,IX26,IX27,IMAX(300),NC,NB,NBR,
     4NCS,NBS,NBRS,
     5COLID1(48),COLID2(48),BMID1(38),BMID2(38),BRID1(20),BRID2(20)
      COMMON ITAB(300),ISFC(300),IFIRST,M,F,DJ(600),                    MAI0002
     1DUM2(1800),DUM3(365100),DUM4(4057),IDUM2(905),
     2IDUM5(1203),IX11,IX12,IX13,IX14,IX15,IDUM6(2),DUM7(130200),
     1NLS,LN,IFLAG,IERR,ITRANS,ISTED
      DIMENSION TITLE(20)                                               MAI0003
      DEFINE FILE 11(600,600,U,IX11)                                    MAI0004
      DEFINE FILE 12(5100,36,U,IX12)                                    MAI0005
      DEFINE FILE 13(576,36,U,IX13)                                     MAI0006
      DEFINE FILE 14(576,36,U,IX14)                                     MAI0007
      DEFINE FILE 15(14,600,U,IX15)                                     MAI0008
      DEFINE FILE 16(48,4,U,IX16)                                       MAI0009
      DEFINE FILE 17(38,4,U,IX17)                                       MAI0010
      DEFINE FILE 18(20,4,U,IX18)                                       MAI0011
      DEFINE FILE 19(300,3,U,IX19)                                      MAI0012
      DEFINE FILE 20(100,3,U,IX20)                                      MAI0013
      DEFINE FILE 21(300,5,U,IX21)                                      MAI0014
      DEFINE FILE 22(300,5,U,IX22)                                      MAI0015
      DEFINE FILE 23(300,5,U,IX23)                                      MAI0016
      DEFINE FILE 24(300,5,U,IX24)                                      MAI0017
      DEFINE FILE 25(300,5,U,IX25)                                      MAI0018
      DEFINE FILE 26(300,5,U,IX26)                                      MAI0019
      DEFINE FILE 27(300,5,U,IX27)                                      MAI0020
C
C  INPUT PROBLEM TITLE
C
      READ(5,5) (TITLE(I),I=1,20)                                       MAI0021
    5 FORMAT(20A4)                                                      MAI0022
      WRITE(6,766)                                                      MAI0023
  766 FORMAT(1H1,"*************************************************MAI0024
     1****************************")
      WRITE(6,6) (TITLE(I),I=1,20)                                      MAI0025
    6 FORMAT(1X,20A4)                                                   MAI0026
      WRITE(6,767)                                                      MAI0027
  767 FORMAT(1X,"**************************************************MAI0028
     1****************************"//)
      READ(5,10) ICOL,IBEAM,IBR,IEXACT,ITOTAL,ITRANS,IUPD,MEMCHG,IFLAG, MAI0029
     1IFLAG1,NLS,NDC,FACT,TOL,EP
   10 FORMAT(12I5,F8.2,2F6.2)                                           MAI0030
      WRITE(6,251) ICOL                                                 MAI0031
```

```
      WRITE(6,252) IBEAM                                                MAI0032
      WRITE(6,253) IBR                                                  MAI0033
      WRITE(6,254) IEXACT                                               MAI0034
      WRITE(6,255) ITOTAL                                               MAI0035
      WRITE(6,256) ITRANS                                               MAI0036
      WRITE(6,257) ILPD                                                 MAI0037
      WRITE(6,258) MEMCHG                                               MAI0038
      WRITE(6,259) IFLAG                                                MAI0039
      WRITE(6,260) IFLAG1                                               MAI0040
      WRITE(6,261) NLS                                                  MAI0041
      WRITE(6,262) NDC                                                  MAI0042
      WRITE(6,263) FACT                                                 MAI0043
      WRITE(6,264) TOL                                                  MAI0044
      WRITE(6,265) EP                                                   MAI0045
  251 FORMAT(1X,"NUMBER OF COLUMNS IN SECTION TABLE --------------------MAI0046
     1----",I5/)
  252 FORMAT(1X,"NUMBER OF BEAMS   IN SECTION TABLE --------------------MAI0047
     1----",I5/)
  253 FORMAT(1X,"NUMBER OF BRACES  IN SECTION TABLE --------------------MAI0048
     1----",I5/)
  254 FORMAT(1X,"NUMBER OF STFDES ANALYSES PER EXACT ANALYSIS-----------MAI0049
     1----",I5/)
  255 FORMAT(1X,"MAXIMUM NUMBER OF ANALYSES  (STFDES + EXACT) ----------MAI0050
     1----",I5/)
  256 FORMAT(1X,"TRANSFORMATION MATRIX PARAMETER (MUST BE 0 OR 1 OR 2) -MAI0051
     1----",I5/)
  257 FORMAT(1X,"MAXIMUM NUMBER OF MEMBER UPDATES PER ANALYSIS ---------MAI0052
     1----",I5/)
  258 FORMAT(1X,"MEMBER CHANGE PARAMETER (MUST BE 0 OR 1) --------------MAI0053
     1----",I5/)
  259 FORMAT(1X,"INITIAL ANALYSIS TYPE PARAMETER (MUST BE 0 OR 1) ------MAI0054
     1----",I5/)
  260 FORMAT(1X,"FINAL   ANALYSIS TYPE PARAMETER *MUST BE 0 OR 1* ------MAI0055
     1----",I5/)
  261 FORMAT(1X,"TOTAL NUMBER OF LOADINGS *P AND Q LOADS,MAX OF 7* -----MAI0056
     1----",I5/)
  262 FORMAT(1X,"NUMBER OF DISPLACEMENT CONSTRAINTS *MAX OF 3* ---------MAI0057
     1----",I5/)
  263 FORMAT(1X,"UNIT LOAD MAGNIFICATION FACTOR ------------------------MAI0058
     2--",F7.2/)
  264 FORMAT(1X,"DISPLACEMENT CONSTRAINT TOLERANCE IN PERCENT ----------MAI0059
     3----",F5.2/)
  265 FORMAT(1X,"INITIAL ERROR IN PERCENT ------------------------------MAI0060
     1----",F5.2/)
      IF(IFLAG.NE.1.AND.IFLAG.NE.0) IERR=1                              MAI0061
      IF(IFLAG1.NE.0.AND.IFLAG1.NE.1) IERR=1                            MAI0062
      IF(MEMCHG.NE.1.AND.MEMCHG.NE.0) IERR=1                            MAI0063
      IF(NDC.LT.1.OR.NDC.GT.3) IERR=1                                   MAI0064
```

```
      IF(NLS.LT.1.OR.NLS.GT.7) IERR=1                                   MAI0065
      IF(ITRANS.NE.0.AND.ITRANS.NE.1.AND.ITRANS.NE.2) IERR=1            MAI0066
      IF(IERR.EQ.1) GO TO 230                                           MAI0067
C
C   INPUT DATA--P,Q TABLE  JOINTS AT WHICH CONSTRAINTS HAVE BEEN MADE
C   COMPONENTS OF DISPLACEMENT AT THESE JOINTS  ACTUAL CONSTRAINT VALUES
C   DISPLACEMENT TYPES
C   INPUT SECTION TABLES
C
      WRITE(6,293)                                                      MAI0068
  293 FORMAT(1H1,"LOCATION AND MAGNITUDE OF DISPLACEMENT CONSTRAINTS")  MAI0069
      WRITE(6,294)                                                      MAI0070
  294 FORMAT(1X,"--------------------------------------------------"//) MAI0071
      DO 20 I=1,NDC                                                     MAI0072
      WRITE(6,295) I                                                    MAI0073
  295 FORMAT(1X,"DISPLACEMENT CONSTRAINT NUMBER",I3)                    MAI0074
      WRITE(6,297)                                                      MAI0075
  297 FORMAT(1X,"---------------------------------"//)                  MAI0076
      READ(5,30) (IGPT(I,J),J=1,6)                                      MAI0077
      READ(5,40) (JT(I,J),J=1,2)                                        MAI0078
      READ(5,50) (XJT(I,J),J=1,4)                                       MAI0079
      READ(5,60) DC(I),IDTYP(I),JFN                                     MAI0080
   30 FORMAT(6I5)                                                       MAI0081
   40 FORMAT(2I5)                                                       MAI0082
   50 FORMAT(4F10.2)                                                    MAI0083
   60 FORMAT(F10.2,2I5)                                                 MAI0084
      IF(IDTYP(I).EQ.0) GO TO 310                                       MAI0085
      IF(XJT(I,1).GT.1.00.OR.XJT(I,2).GT.1.00.OR.XJT(I,3).GT.1.00)      MAI0086
     1IERR=1
      WRITE(6,300) JT(I,1),DC(I)                                        MAI0087
  300 FORMAT(1X,"TRANSLATIONAL DISPLACEMENT CONSTRAINT AT JOINT",I5,2X, MAI0088
     1"IS",F6.2,2X,"INCHES"/)
      WRITE(6,305) XJT(I,1),XJT(I,2),XJT(I,3)                           MAI0089
  305 FORMAT(1X,"THE EXACT DIRECTION OF DISPLACEMENT IS ",F4.2,"X, ",   MAI0090
     2F4.2,"Y, ",F4.2,"Z"/)
      WRITE(6,18) IGPT(I,1)                                             MAI0091
   18 FORMAT(1X,"THE UNIT LOAD FOR THIS CASE IS LOADING NO.",I5/)       MAI0092
      WRITE(6,19)                                                       MAI0093
   19 FORMAT(1X,"*******************************************************MAI0094
     1******************************************************************
     2**********"//)
      GO TO 20                                                          MAI0095
  310 WRITE(6,315) JFN,DC(I)                                            MAI0096
  315 FORMAT(1X,"ROTATIONAL DISPLACEMENT CONSTRAINT ON FLOOR",I5,2X,"IS"MAI0097
     3,F6.2,2X,"RADIANS"/)
      WRITE(6,320) JT(I,1),JT(I,2)                                      MAI0098
  320 FORMAT(1X,"A CHORD BETWEEN JOINT",I5,2X,"AND JOINT",I5,2X,"WILL BEMAI0099
     4 USED TO MEASURE ROTATION"/)
```

```
      WRITE(6,325) JT(I,1),XJT(I,1),XJT(I,2)
  325 FORMAT(1X,"THE X AND Z COORDINATES OF JOINT",I5,2X,"ARE *",
     5F8.2,",",F8.2,"*"/)
      WRITE(6,325) JT(I,2),XJT(I,3),XJT(I,4)
      WRITE(6,18) ICPT(I,1)
      WRITE(6,19)
      GO TO 20
   20 CONTINUE
      IF(IERR.EQ.1) GO TO 230
      CALL PTABLE
C
C  INITIALIZE COUNTER VALUES TO ZERO
C
      IFIRST=0
      ICHK=0
      JCHK=0
      IERR=0
      NCS=0
      NPS=0
      NPRS=0
      ITER=0
      IREG=0
      ISTFD=0
      DO 67 I=1,300
      IMAX(I)=0
   67 CONTINUE
   70 IF(JCHK.EQ.(ITOTAL-1)) IFLAG=IFLAG1
      CALL MODFR2
      IF(IERR.EQ.1) GO TO 235
      DO 73 I=1,NDC
      IOK(I)=1
   73 CONTINUE
      IF(IFLAG.EQ.0) GO TO 75
      ICHK=ICHK+1
      ISTFD=ISTFD+1
      GO TO 77
   75 ICHK=0
      IREG=IREG+1
   77 JCHK=JCHK+1
C
C  CHECK TO SEE IF DISPLACEMENT CONSTRAINTS ARE SATISFIED
C
      WRITE(6,900)
  900 FORMAT(1H1,"  DC    * P LOAD *    COMPUTED    *    VALUE OF")
      WRITE(6,901)
  901 FORMAT(1X,"NUMBER * NUMBER * DISPLACEMENT * DISP CONSTR.")
      WRITE(6,902)
  902 FORMAT(1X,"**************************************************"/)
```

```
      DO 80 I=1,NDC
      NPL=IQPT(I,2)+2
      DO 80 INP=3,NPL
      J=IQPT(I,INP)
      WRITE(6,92) I,J,DFLEX(I,J),DC(I)
   92 FORMAT(1X,I4,I9,5X,1PE12.5,3X,1PE12.5)
      IF(ABS(DELEX(I,J)).GT.(DC(I)*(1.0+(TOL/100.0)))) IOK(I)=0
   80 CONTINUE
      DO 90 I=1,NDC
      IF(IOK(I).EQ.0) GO TO 100
   90 CONTINUE
      GO TO 225
  100 DO 110 I=1,NDC
      IMEM(I)=0
  110 CONTINUE
C
C   CALCULATE ERROR TERM
C
      DO 120 I=1,NDC
      NPL=IGPT(I,2)+2
      DO 120 JNP=3,NPL
      J=IQPT(I,JNP)
      IF(IFIRST.EQ.0) GO TO 115
      DERROR(I,J)=(DELEX(I,J)-DVW(I,J))
      GO TO 120
  115 DERROR(I,J)=DELEX(I,J)*EP/100.0
      GO TO 120
  120 CONTINUE
      IF(IFIRST.EQ.0) GO TO 130
      GO TO 150
  130 IFIRST=1
      DO 140 I=1,NDC
      QDMIN(I)=10000.0
      MOD(I)=1
  140 CONTINUE
  150 JMEM=0
      NC=0
      NB=0
      NBR=0
      WRITE(6,897)
  897 FORMAT(1X,//," ")
      WRITE(6,920)
  920 FORMAT(1X,"  DC    * P LOAD *  APPROXIMATE *  ERROR       *   VALUE
     1 OF    * NO. OF")
      WRITE(6,921)
  921 FORMAT(1X,"NUMBER * NUMBER * DISPLACEMENT *        TERM   * DISP CO
     1NSTR. * INC MEM")
      WRITE(6,922)
```

```
  922 FORMAT(1X,"*****************************************************************
     1****************"/)
C
C  CALL SUBROUTINES TO CALCULATE SENSITIVITY COEFFICIENTS AND TO INCREASE
C
C  MEMBER SIZES WHERE APPROPRIATE
C
  160 CALL DSENCO
      CALL MDCHG
      CALL DVWCHG
      IF(IERR.EQ.1) GO TO 235
      DO 165 I=1,NDC
      IOK(I)=1
  165 CONTINUE
C
C  CHECK TO SEE IF DISPLACEMENT CONSTRAINTS ARE SATISFIED
C
      DO 170 I=1,NDC
      NPL=IGPT(I,2)+2
      DO 170 INP=3,NPL
      J=IGPT(I,INP)
      WRITE(6,84) I,J,DVW(I,J),DERROR(I,J),DC(I),JMEM
   84 FORMAT(1X,I4,I9,5X,1PE12.5,3X,1PE12.5,3X,1PE12.5,I7)
      IF(ABS(DVW(I,J)+DERROR(I,J)).GT.(DC(I)*(1.0-(TOL/100.0))))
     1IOK(I)=0
  170 CONTINUE
      DO 175 I=1,NDC
      IF(IOK(I).EQ.0) GO TO 200
  175 CONTINUE
C
C  FIND OUT HOW MANY AND WHICH TYPE OF ANALYSES HAVE BEEN PERFORMED
C  TO DETERMINE WHICH BRANCH TO TAKE
C
  180 IF(JCHK.EQ.ITOTAL) GO TO 210
      NCS=NCS+NC
      NBS=NBS+NB
      NBRS=NBRS+NBR
      WRITE(6,400) JCHK
  400 FORMAT(1H1,"MEMBER CHANGE DATA FOR ANALYSIS NUMBER",I5)
      WRITE(6,405)
  405 FORMAT(1X,"------------------------------------------------"//)
      IF(IFLAG.EQ.0) GO TO 700
      WRITE(6,710)
  710 FORMAT(1X,"TYPE OF ANALYSIS --- STFDES"/)
      GO TO 730
  700 WRITE(6,720)
  720 FORMAT(1X,"TYPE OF ANALYSIS --- REGULAR"/)
      GO TO 730
```

```
730 WRITE(6,178) NC
178 FORMAT(1X,"NUMBER OF COLUMNS CHANGED -----------",I5/)
    WRITE(6,179) NB
179 FORMAT(1X,"NUMBER OF BEAMS   CHANGED ----- -----",I5/)
    WRITE(6,181) NBR
181 FORMAT(1X,"NUMBER OF BRACES  CHANGED -----------",I5/)
    WRITE(6,182) JMEM
182 FORMAT(1X,"TOTAL NUMBER OF MEMBERS CHANGED -----",I5//)
    WRITE(6,600)
600 FORMAT(1X,"CURRENT MATERIAL WEIGHT AND COST EVALUATION")
    WRITE(6,601)
601 FORMAT(1X,"-------------------------------------------"//)
    CALL COSTWT
    WRITE(6,602)
602 FORMAT(1X,"DISPLACEMENT DATA")
    WRITE(6,603)
603 FORMAT(1X,"-----------------"//)
    WRITE(6,607)
607 FORMAT(1X,"  DC   * P LOAD *   COMPUTED   *  APPROXIMATE *   VALUE
   1 OF")
    WRITE(6,608)
608 FORMAT(1X,"NUMBER * NUMBER * DISPLACEMENT * DISPLACEMENT * DISP CO
   1NSTR.")
    WRITE(6,609)
609 FORMAT(1X,"*****************************************************
   1*****"/)
    DO 604 I=1,NDC
    NPL=IQPT(I,2)+2
    DO 604 IJ=3,NPL
    J=IQPT(I,IJ)
    WRITE(6,605) I,J,DELFY(I,J),DVW(I,J),DC(I)
605 FORMAT(1X,I4,I9,5X,1PE12.5,3X,1PE12.5,3X,1PE12.5)
604 CONTINUE
    IF(ICHK.EQ.IEXACT) GO TO 190
    IFLAG=1
    GO TO 70
190 IFLAG=0
    GO TO 70
200 IF(JMEM.GE.IUFD) GO TO 180
    GO TO 160
210 NCS=NCS+NC
    NBS=NBS+NB
    NBRS=NBRS+NBR
    WRITE(6,400) JCHK
    WRITE(6,405)
    IF(IFLAG.EQ.0) GO TO 800
    WRITE(6,710)
    GO TO 830
```

```
800 WRITE(6,720)
    GO TO 830
830 WRITE(6,178) NC
    WRITE(6,179) NB
    WRITE(6,181) NBR
    WRITE(6,182) JMEM
    WRITE(6,602)
    WRITE(6,603)
    WRITE(6,607)
    WRITE(6,608)
    WRITE(6,609)
    DO 960 I=1,NDC
    NPL=IGPT(I,2)+2
    DO 960 IJ=3,NPL
    J=IGPT(I,IJ)
    WRITE(6,605) I,J,DELEX(I,J),DVW(I,J),DC(I)
960 CONTINUE
    WRITE(6,897)
    WRITE(6,224)
224 FORMAT(1X,"***************************************************************
   1**")
    WRITE(6,221)
221 FORMAT(1X,"MAXIMUM NUMBER OF ANALYSES PERFORMED -- DESIGN INCOMPLE
   1TE")
    WRITE(6,224)
225 CALL OUTPUT
    GO TO 235
230 WRITE(6,231)
231 FORMAT(1X,"ALL INPUT DATA ARE NOT WITHIN ALLOWED LIMITS...CHECK"/)
235 STOP
    END
```

```
*DECK PTABLE                                                            PTA0000
      SUBROUTINE PTABLE                                                 PTA0001
      COMMON Q(300,3),DELEX(3,6),DERRCR(3,6),DVW(3,6),DELVWN(3,6),      PTA0002
     1DELVW(3,6),XJT(3,4),DC(3),QDMIN(3),THETA(2),FACT,IREG,JCHK,
     2IGPT(3,6),JT(3,2),IDTYP(3),ICK(3),IMEM(3),MOD(3),ICHG(3),IDC(3),
     3ICOL,IBEAM,IBR,NDC,ITER,IDERIV,MEMCHG,IN,JMEM,LL,T,IX16,IX17,IX18,
     4IX19,IX20,IX21,IX22,IX23,IX24,IX25,IX26,IX27,IMAX(300),NC,NB,NBR,
     4NCS,NBS,NBRS,
     5COLID1(48),COLID2(48),BMID1(38),BMID2(38),BRID1(20),BRID2(20)
      COMMON ITAB(300),ISEC(300),IFIRST,M,F,DJ(600),                    PTA0003
     1DUM2(1800),DUM3(365100),DUM4(4057),IDUM2(905),
     2IDUM5(1203),IX11,IX12,IX13,IX14,IX15,IDUM6(2),DUM7(130200),
     1NLS,LN,IFLAG,IERR,ITRANS,ISTED
C
C  INPUT SECTION TABLES--COLUMNS BEAMS BRACES IN THAT ORDER
C
      WRITE(6,300) ICOL                                                 PTA0004
  300 FORMAT(1H1,"COLUMN SECTION TABLE WITH",I5,2X,"MEMBERS")           PTA0005
      WRITE(6,305)                                                      PTA0006
  305 FORMAT(1X,"----------------------------------------"//)           PTA0007
      WRITE(6,75)                                                       PTA0008
   75 FORMAT(1X,"NUMBER     NAME      AX        IX        IY        IZ")PTA0009
      DO 680 I=1,ICOL                                                   PTA0010
      READ(5,650) J,COLID1(J),COLID2(J),AREA,RIX,RIY,RIZ                PTA0011
  650 FORMAT(8X,I7,T1,2A4,7X,4F10.2)                                    PTA0012
      WRITE(6,15) J,COLID1(J),COLID2(J),AREA,RIX,RIY,RIZ                PTA0013
   15 FORMAT(1X,I4,3X,2A4,F8.2,3F9.2)                                   PTA0014
      WRITE(16"J) AREA,RIX,RIY,RIZ                                      PTA0015
  680 CONTINUE                                                          PTA0016
      WRITE(6,310) IBEAM                                                PTA0017
  310 FORMAT(1H1,"BEAM SECTION TABLE WITH",I5,2X,"MEMBERS")             PTA0018
      WRITE(6,315)                                                      PTA0019
  315 FORMAT(1X,"----------------------------------------"//)           PTA0020
      WRITE(6,75)                                                       PTA0021
      DO 500 I=1,IBEAM                                                  PTA0022
      READ(5,650) J,BMID1(J),BMID2(J),AREA,RIX,RIY,RIZ                  PTA0023
      WRITE(6,15) J,BMID1(J),BMID2(J),AREA,RIX,RIY,RIZ                  PTA0024
      WRITE(17"J) AREA,RIX,RIY,RIZ                                      PTA0025
  500 CONTINUE                                                          PTA0026
      IF(IBR.EQ.0) GO TO 700                                            PTA0027
      WRITE(6,320) IBR                                                  PTA0028
  320 FORMAT(1H1,"BRACE SECTION TABLE WITH",I5,2X,"MEMBERS")            PTA0029
      WRITE(6,325)                                                      PTA0030
  325 FORMAT(1X,"----------------------------------------"//)           PTA0031
      WRITE(6,75)                                                       PTA0032
      DO 550 I=1,IBR                                                    PTA0033
      READ(5,650) J,BRID1(J),BRID2(J),AREA,RIX,RIY,RIZ                  PTA0034
      WRITE(6,15) J,BRID1(J),BRID2(J),AREA,RIX,RIY,RIZ                  PTA0035
```

```
      WRITE(18'J) AREA,RIX,RIY,RIZ
  550 CONTINUE
  700 RETURN
      END
```

```
*DECK TSCHK                                                            TSC0000
      SUBROUTINE TSCHK                                                 TSC0001
      COMMON Q(300,3),DELEX(3,6),DERROR(3,6),DVW(3,6),DELVWN(3,6),     TSC0002
     1DELVW(3,6),XJT(3,4),DC(3),QDMIN(3),THETA(2),FACT,IREG,JCHK,
     2IRPT(3,6),JT(3,2),IDTYP(3),IOK(3),IMEM(3),MOD(3),ICHG(3),IDC(3),
     3ICOL,IBEAM,IBR,NDC,ITER,IDERIV,MEMCHG,IN,JMEM,LL,I,IX16,IX17,IX18,
     4IX19,IX20,IX21,IX22,IX23,IX24,IX25,IX26,IX27,IMAX(300),NC,NB,NBR,
     4NCS,NBS,NBRS,
     5COLID1(48),COLID2(48),BMID1(38),BMID2(38),BRID1(20),BRID2(20)
      COMMON ITAB(300),ISEC(300),IFIRST,M,E,DJ(400),                   TSC0003
     1DUM2(1800),DUM3(305100),DUM4(4057),IDUM2(905),
     2IDUM5(1203),IX11,IX12,IX13,IX14,IX15,IDUM6(2),DUM7(130200),
     1NLS,LN,IFLAG,IERR,ITRANS,ISTEO
C
C   IF A MEMBER IS AT ITS MAXIMUM SIZE SET ITS DISPLACEMENT SENSITIVITY
C   COEFFICIENT TO 10000.  ALSO MAKE SURE THE PROPER TABLE IS SPECIFIED
C
      DO 730 I=1,M                                                     TSC0004
      IF(ITAB(I).EQ.16) GO TO 660                                      TSC0005
      IF(ITAB(I).EQ.17) GO TO 670                                      TSC0006
      IF(ITAB(I).EQ.18) GO TO 680                                      TSC0007
  650 IERR=1                                                           TSC0008
      GO TO 710                                                        TSC0009
  660 IF(ISEC(I).GE.ICOL) GO TO 690                                    TSC0010
      IF(ISEC(I).LE.0) GO TO 650                                       TSC0011
      GO TO 730                                                        TSC0012
  670 IF(ISEC(I).GE.IBEAM) GO TO 690                                   TSC0013
      IF(ISEC(I).LE.0) GO TO 650                                       TSC0014
      GO TO 730                                                        TSC0015
  680 IF(ISEC(I).GE.IBR) GO TO 690                                     TSC0016
      IF(ISEC(I).LE.0) GO TO 650                                       TSC0017
      GO TO 730                                                        TSC0018
  690 IF(ITAB(I).EQ.16.AND.ISEC(I).GT.ICOL) GO TO 650                  TSC0019
      IF(ITAB(I).EQ.17.AND.ISEC(I).GT.IBEAM) GO TO 650                 TSC0020
      IF(ITAB(I).EQ.18.AND.ISEC(I).GT.IBR) GO TO 650                   TSC0021
      IMAX(I)=1                                                        TSC0022
      DO 700 K=1,NDC                                                   TSC0023
      Q(I,K)=10000.0                                                   TSC0024
  700 CONTINUE                                                         TSC0025
      GO TO 730                                                        TSC0026
  710 WRITE(6,720) I,ITAB(I),ISEC(I)                                   TSC0027
  720 FORMAT(1X,"IMPROPER MEMBER TABLE OR SECTION SPECIFIED---MEMBER=", TSC0028
     1I5,3X,"TABLE=",I3,3X,"SECTION=",I4/)
  730 CONTINUE                                                         TSC0029
      RETURN                                                           TSC0030
      END                                                              TSC0031
```

```
*DECK DSENCO                                                            DSE0000
      SUBROUTINE DSENCO                                                 DSE0001
      COMMON Q(300,3),DELEX(3,6),DERROR(3,6),DVW(3,6),DELVWN(3,6),      DSE0002
     1DELVW(3,6),XJT(3,4),DC(3),QDMIN(3),THETA(2),FACT,IREG,JCHK,
     2IGPT(3,6),JT(3,2),IDTYP(3),IOK(3),IMEM(3),MOD(3),ICHG(3),IDC(3),
     3ICOL,IBEAM,IBR,NDC,ITER,IDERIV,MEMCHG,IN,JMEM,LL,I,IX16,IX17,IX18,
     4IX19,IX20,IX21,IX22,IX23,IX24,IX25,IX26,IX27,IMAX(300),NC,NB,NBR,
     4NCS,NBS,NBRS,
     5COLID1(48),COLID2(48),BMID1(38),BMID2(38),BRID1(20),BRID2(20)
      COMMON ITAB(300),ISEC(300),IFIRST,M,E,FJ(600),                    DSE0003
     1DUM2(1800),DUM3(365100),DUM4(4057),IDUM2(905),
     2IDUM5(1203),IX11,IX12,IX13,IX14,IX15,IDUM6(2),DUM7(130200),
     1NLS,LN,IFLAG,IERR,ITRANS,ISTED
C
C   CALCULATE DISPLACEMENT SENSITIVITY COEFFICIENT AND FOR EACH DISPLACEM
C                                                                   ENT
C   CONSTRAINT FIND THE MINIMUM OF THESE VALUES AND THE MEMBER
C   ASSOCIATED WITH THEM
C
      DO 270 LL=1,NDC                                                   DSE0004
      IF(IOK(LL).NE.0) GO TO 270                                        DSE0005
      DO 250 I=1,M                                                      DSE0006
      IF(IMAX(I).EQ.1) GO TO 240                                        DSE0007
      IF(IMEM(LL).NE.0) GO TO 240                                       DSE0008
      IDERIV=1                                                          DSE0009
      CALL VIRTWK                                                       DSE0010
  240 IF(Q(I,LL).GT.QDMIN(LL)) GO TO 250                                DSE0011
      QDMIN(LL)=Q(I,LL)                                                 DSE0012
      MOD(LL)=I                                                         DSE0013
  250 CONTINUE                                                          DSE0014
      IF(IMEM(LL).NE.0) GO TO 270                                       DSE0015
      IMEM(LL)=1                                                        DSE0016
      NPL=IGPT(LL,2)+2                                                  DSE0017
      DO 260 KNP=3,NPL                                                  DSE0018
      J=IGPT(LL,KNP)                                                    DSE0019
      DVW(LL,J)=DELEX(LL,J)                                             DSE0020
  260 CONTINUE                                                          DSE0021
  270 CONTINUE                                                          DSE0022
      RETURN                                                            DSE0023
      END                                                               DSE0024
```

```
*DECK NDCHG                                                                NDC0000
      SUBROUTINE NDCHG                                                     NDC0001
      COMMON Q(300,3),DELEX(3,6),DERROR(3,6),DVW(3,6),DELVWN(3,6),         NDC0002
     1DELVW(3,6),XJT(3,4),DC(3),QDMIN(3),THETA(2),FACT,IREG,JCHK,
     2IQPT(3,6),JT(3,2),IDTYP(3),IOK(3),IMEM(3),MQD(3),ICHG(3),IDC(3),
     3ICOL,IBEAM,IBR,NDC,ITER,IDERIV,MEMCHG,IN,JMEM,LL,I,IX16,IX17,IX18,
     4IX19,IY20,IX21,IX22,IX23,IX24,IX25,IX26,IX27,IMAX(300),NC,NB,NBR,
     4NCS,NBS,NBRS,
     5COLID1(48),COLID2(48),BMID1(38),BMID2(38),BRID1(20),BRID2(20)
      COMMON ITAB(300),ISEC(300),IFIRST,M,E,DJ(600),                       NDC0003
     1DUM2(1800),DUM3(365100),DUM4(4057),IDUM2(905),
     2IDUM5(1203),IX11,IX12,IX13,IX14,IX15,IDUM6(2),DUM7(130200),
     1NLS,LN,IFLAG,IERR,ITRANS,ISTED
C
C   DETERMINE WHICH DISPLACEMENT DIRECTION WILL HAVE A MEMBER CHANGE
C   ASSOCIATED WITH IT
C
      DO 280 I=1,NDC                                                       NDC0004
      ICHG(I)=0                                                            NDC0005
  280 CONTINUE                                                             NDC0006
      IF(MEMCHG.EQ.1) GO TO 320                                            NDC0007
      XMINQD=10000.0                                                       NDC0008
      DO 300 I=1,NDC                                                       NDC0009
      IF(IOK(I).NE.0) GO TO 300                                            NDC0010
      IF(QDMIN(I).GT.XMINQD) GO TO 300                                     NDC0011
      XMINQD=QDMIN(I)                                                      NDC0012
      IDIR=I                                                               NDC0013
  300 CONTINUE                                                             NDC0014
      DO 310 I=1,NDC                                                       NDC0015
      IF(I.EQ.IDIR) GO TO 310                                              NDC0016
      ICHG(I)=1                                                            NDC0017
  310 CONTINUE                                                             NDC0018
      GO TO 350                                                            NDC0019
  320 DO 340 J=1,NDC                                                       NDC0020
      IF(IOK(J).NE.0) ICHG(J)=1                                            NDC0021
      IF(J.EQ.NDC) GO TO 340                                               NDC0022
      L1=J+1                                                               NDC0023
      DO 330 LL=L1,NDC                                                     NDC0024
      IF(MQD(J).NE.MQD(LL)) GO TO 330                                      NDC0025
      IF(ICHG(J).NE.0) GO TO 330                                           NDC0026
      ICHG(LL)=1                                                           NDC0027
  330 CONTINUE                                                             NDC0028
  340 CONTINUE                                                             NDC0029
  350 RETURN                                                               NDC0030
      END                                                                  NDC0031
```

```
*DECK DVWCHG                                                                  DVW0000
      SUBROUTINE DVWCHG                                                       DVW0001
      COMMON Q(300,3),DELEX(3,6),DERROR(3,6),DVW(3,6),DELVWN(3,6),            DVW0002
     1DELVW(3,6),XJT(3,4),DC(3),ODMIN(3),THETA(2),FACT,IREG,JCHK,
     2IOPT(3,6),JT(3,2),IDTYP(3),ICK(3),IMEM(3),MOD(3),ICHG(3),IDC(3),
     3ICOL,IBEAM,IBR,NDC,ITER,IDERIV,MEMCHG,IN,JMEM,LL,I,IX16,IX17,IX18,
     4IX19,IX20,IX21,IX22,IX23,IX24,IX25,IX26,IX27,IMAX(300),NC,NB,NBR,
     4NCS,NBS,NBRS,
     5COLID1(48),COLID2(48),BMID1(38),BMID2(38),BRID1(20),BRID2(20)
      COMMON ITAB(300),ISEC(300),IFIRST,M,E,DJ(600),                          DVW0003
     1DUM2(1800),DUM3(365100),DUM4(4057),IDUM2(905),
     2IDUM5(1203),IX11,IX12,IX13,IX14,IX15,IDUM6(2),DUM7(130200),
     1NLS,LN,IFLAG,IERR,ITRANS,ISTFD
C
C  CALCULATE BY VIRTUAL WORK THE DISPLACEMENT CONTRIBUTION OF MEMBER I
C
      DO 400 K=1,NDC                                                          DVW0004
      IF(ICHG(K).EQ.1) GO TO 400                                              DVW0005
      I=MOD(K)                                                                DVW0006
      IDERIV=0                                                                DVW0007
      IN=0                                                                    DVW0008
      DO 360 LL=1,NDC                                                         DVW0009
      CALL VIRTWK                                                             DVW0010
  360 CONTINUE                                                                DVW0011
C
C  INCREASE MEMBER I IN SIZE BY ONE IN SECTION TABLE
C
      CALL INCMEM                                                             DVW0012
      IF(IERR.EQ.1) GO TO 410                                                 DVW0013
      JMEM=JMEM+1                                                             DVW0014
      ITER=ITER+1                                                             DVW0015
      IN=1                                                                    DVW0016
C
C  CALCULATE NEW CONTRIBUTION OF MEMBER I
C
      DO 370 LL=1,NDC                                                         DVW0017
      CALL VIRTWK                                                             DVW0018
  370 CONTINUE                                                                DVW0019
C
C  CALCULATE THE NEW VALUE OF DISPLACEMENT BY ADDING THE REDUCTION DUE T
C                                                                   O
C  INCREASE OF A MEMBER
C
      DO 380 LL=1,NDC                                                         DVW0020
      NPL=IOPT(LL,2)+2                                                        DVW0021
      DO 380 LNP=3,NPL                                                        DVW0022
      J=IOPT(LL,LNP)                                                          DVW0023
      DCH=DELVWN(LL,J)-DELVW(LL,J)                                            DVW0024
```

```
      IF(DVW(LL,J).GT..000001.AND.DCH.GT..000001) GO TO 375
      IF(DVW(LL,J).LT..000001.AND.DCH.LT..000001) GO TO 375
      DVW(LL,J)=DVW(LL,J)+(DCH/FACT)
      GO TO 380
  375 DVW(LL,J)=DVW(LL,J)-(DCH/FACT)
  380 CONTINUE
      IF(IMAX(I).EQ.1) GO TO 395
      IDERIV=1
C
C  CALCULATE NEW SENSITIVITY FACTOR FOR MEMBER WHICH WAS CHANGED
C
      DO 390 LL=1,NDC
      CALL VIRTWK
  390 CONTINUE
  395 QDMIN(K)=Q(I,K)
  400 CONTINUE
  410 RETURN
      END
```

```
*DECK INCMEM                                                                  INC0000
      SUBROUTINE INCMEM                                                       INC0001
      COMMON Q(300,3),DELEX(3,6),DERROP(3,6),DVW(3,6),DELVWN(3,6),           INC0002
     1DELVW(3,6),XJT(3,4),DC(3),QDMIN(3),THETA(2),FACT,IREG,JCHK,
     2ICPT(3,6),JT(3,2),ICTYP(3),ICK(3),IMEM(3),MOD(3),ICHG(3),IDC(3),
     3ICOL,IBEAM,IBR,NDC,ITER,IDERIV,MEMCHG,IN,JMEM,LL,I,IX16,IX17,IX18,
     4IX19,IX20,IX21,IX22,IX23,IX24,IX25,IX26,IX27,IMAX(300),NC,NB,NBR,
     4NCS,NBS,NBRS,
     5COLID1(48),COLID2(48),BMID1(38),BMID2(38),BRID1(20),BRID2(20)
      COMMON ITAB(300),ISEC(300),IFIRST,M,F,DJ(600),                         INC0003
     1DUM2(1800),DUM3(365100),DUM4(4057),IDUM2(905),
     2IDUM5(1203),IX11,IX12,IX13,IX14,IX15,IDUM6(2),DUM7(130200),
     1NLS,LN,IFLAG,IERR,ITRANS,ISTED
C
C     INCREASE MEMBER I IN SIZE BY ONE IN SECTION TABLE
C
      DO 480 KL=1,2                                                           INC0004
      IF(ITAB(I).EQ.16) GO TO 420                                             INC0005
      IF(ITAB(I).EQ.17) GO TO 460                                             INC0006
      IF(ITAB(I).EQ.18) GO TO 470                                             INC0007
  415 IERR=1                                                                  INC0008
      GO TO 490                                                               INC0009
  420 IF(ISEC(I).GE.ICOL) GO TO 440                                           INC0010
      IF(KL.EQ.1) NC=NC+1                                                     INC0011
  430 IF(KL.NE.1) GO TO 480                                                   INC0012
      ISEC(I)=ISEC(I)+1                                                       INC0013
      WRITE(6,235) I,ITAB(I),ISEC(I)                                          INC0014
  235 FORMAT(1X,/,1X,"MEMBER CHANGED,",I5,2X,"TABLE,",I5,2X,"NEW SECTIONINC0015
     1 NO.,",I5/)
      GO TO 480                                                               INC0016
  440 IF(ITAB(I).EQ.16.AND.ISEC(I).GT.ICOL) GO TO 415                         INC0017
      IF(ITAB(I).EQ.17.AND.ISEC(I).GT.IBEAM) GO TO 415                        INC0018
      IF(ITAB(I).EQ.18.AND.ISEC(I).GT.IBR) GO TO 415                          INC0019
      IF(KL.EQ.1) GO TO 485                                                   INC0020
      IMAX(I)=1                                                               INC0021
      DO 450 J=1,NDC                                                          INC0022
      Q(I,J)=10000.0                                                          INC0023
  450 CONTINUE                                                                INC0024
      GO TO 480                                                               INC0025
C
C     CHECK TO SEE IF A MEMBER IS AT ITS MAXIMUM SIZE
C
  460 IF(ISEC(I).GE.IBEAM) GO TO 440                                          INC0026
      IF(KL.EQ.1) NB=NB+1                                                     INC0027
      GO TO 430                                                               INC0028
  470 IF(ISEC(I).GE.IBR) GO TO 440                                            INC0029
      IF(KL.EQ.1) NBR=NBR+1                                                   INC0030
      GO TO 430                                                               INC0031
```

```
480 CONTINUE
    GO TO 500
485 IERR=1
    WRITE(6,487)
487 FORMAT(1X,/,"ALL MEMBERS ARE AT MAXIMUM SIZE"/)
    GO TO 500
490 WRITE(6,495) I,ITAB(I),ISEC(I)
495 FORMAT(1X,"IMPROPER MEMBER TABLE OR SECTION SPECIFIED---MEMBER=",
   1I5,3X,"TABLE=",I3,3X,"SECTION=",I4/)
500 RETURN
    END
```

```
*DECK DEXACT                                                          DEX0000
      SUBROUTINE DEXACT                                               DEX0001
      COMMON Q(300,3),DELFX(3,6),DERROR(3,6),DVW(3,6),DELVWN(3,6),    DEX0002
     1DELVW(3,6),XJT(3,4),DC(3),CDMIN(3),THETA(2),FACT,IREG,JCHK,
     2IGPT(3,6),JT(3,2),IDTYP(3),ICK(3),IMEM(3),MOD(3),ICHG(3),IDC(3),
     3ICOL,IBEAM,IBR,NDC,ITER,IDERIV,MEMCHG,IN,JMEM,LL,I,IX16,IX17,IX18,
     4IX19,IX20,IX21,IX22,IX23,IX24,IX25,IX26,IX27,IMAX(300),NC,NB,NBR,
     4NCS,NBS,NBRS,
     5COLID1(48),COLID2(48),BMID1(38),BMID2(38),BRID1(20),BRID2(20)
      COMMON ITAB(300),ISEC(300),IFIRST,M,E,DJ(600),                  DEX0003
     1DUM2(1800),DUM3(365100),DUM4(4057),IDUM2(905),
     2IDUM5(1203),IX11,IX12,IX13,IX14,IX15,IDUM6(2),DUM7(130200),
     1NLS,LN,IFLAG,IERR,ITRANS,ISTED
      DO 510 I=1,NDC                                                  DEX0004
      IDC(I)=0                                                        DEX0005
  510 CONTINUE                                                        DEX0006
      DO 520 I=1,NDC                                                  DEX0007
      NPL=IGPT(I,2)+2                                                 DEX0008
      DO 520 J=3,NPL                                                  DEX0009
      IF(LN.EQ.IGPT(I,J)) IDC(I)=1                                    DEX0010
  520 CONTINUE                                                        DEX0011
      DO 530 I=1,NDC                                                  DEX0012
      IF(IDC(I).EQ.1) GO TO 540                                       DEX0013
  530 CONTINUE                                                        DEX0014
      GO TO 610                                                       DEX0015
  540 IF(IDTYP(I).EQ.0) GO TO 550                                     DEX0016
C
C   CALCULATE EXACT DISPLACEMENT FOR A TRANSLATIONAL TYPE DISPLACEMENT
C
      DX=DJ(6*JT(I,1)-5)                                              DEX0017
      DY=DJ(6*JT(I,1)-4)                                              DEX0018
      DZ=DJ(6*JT(I,1)-3)                                              DEX0019
      DELEX(I,LN)=XJT(I,1)*DX+XJT(I,2)*DY+XJT(I,3)*DZ                 DEX0020
      GO TO 530                                                       DEX0021
C
C   CALCULATE EXACT DISPLACEMENT FOR A ROTATIONAL TYPE DISPLACEMENT
C
  550 JT1=JT(I,1)                                                     DEX0022
      XCO1=XJT(I,1)                                                   DEX0023
      ZCO1=XJT(I,2)                                                   DEX0024
      JT2=JT(I,2)                                                     DEX0025
      XCO2=XJT(I,3)                                                   DEX0026
      ZCO2=XJT(I,4)                                                   DEX0027
      DX1=DJ(6*JT1-5)                                                 DEX0028
      DZ1=DJ(6*JT1-3)                                                 DEX0029
      DX2=DJ(6*JT2-5)                                                 DEX0030
      DZ2=DJ(6*JT2-3)                                                 DEX0031
      INX=0                                                           DEX0032
```

```
      INZ=0
      DO 600 J=1,2
      XL1=XC02-XC01
      ZL1=ZC02-ZC01
      IF(XL1.LE.-.0001) INX=1
      IF(ZL1.LT..0000) INZ=1
      IF(XL1.GT.-.0001.AND.XL1.LT..0001) GO TO 560
      A=ZL1/XL1
      B=ATAN(A)
      IF(INX.EQ.1) GO TO 580
      IF(INZ.EQ.1) GO TO 585
555   THETA(J)=B
557   IF(J.NE.1) GO TO 590
      XC01=XC01+DX1
      XC02=XC02+DX2
      ZC01=ZC01+DZ1
      ZC02=ZC02+DZ2
      INX=0
      INZ=0
      GO TO 600
560   IF(INZ.EQ.1) GO TO 570
      B=1.570796327
      GO TO 555
570   B=4.71238898
      GO TO 555
580   THETA(J)=B+3.141592654
      GO TO 557
585   THETA(J)=B+6.283185307
      GO TO 557
590   DELEX(I,LN)=THETA(2)-THETA(1)
      IF(DELEX(I,LN).GT.1.570796327) GO TO 593
      GO TO 530
593   IF(DELEX(I,LN).GT.3.141592654) GO TO 595
      DELEX(I,LN)=3.141592654-DELEX(I,LN)
      GO TO 530
595   IF(DELEX(I,LN).GT.4.71238898) GO TO 597
      DELEX(I,LN)=DELEX(I,LN)-3.141592654
      GO TO 530
597   DELEX(I,LN)=DELEX(I,LN)-6.28318307
      GO TO 530
600   CONTINUE
      GO TO 530
610   RETURN
      END
```

```
*DECK VIRTWK                                                                  VIR0000
      SUBROUTINE VIRTWK                                                        VIR0001
      COMMON Q(300,3),DELEX(3,6),DERROR(3,6),DVW(3,6),DELVWN(3,6),            VIR0002
     1DELVW(3,6),XJT(3,4),DC(3),QDMIN(3),THETA(2),FACT,IREG,JCHK,
     2IGPT(3,6),JT(3,2),IDTYP(3),IOK(3),IMEM(3),MOD(3),ICHG(3),IDC(3),
     3ICOL,IBEAM,IBR,NDC,ITER,IDERIV,MEMCHG,IN,JMEM,LL,I,IX16,IX17,IX18,
     4IX19,IX20,IX21,IX22,IX23,IY24,IX25,IY26,IX27,IMAX(300),NC,NB,NBR,
     4NCS,NBS,NBRS,
     5COLID1(48),COLID2(48),BMID1(38),BMID2(38),BRID1(20),BRID2(20)
      COMMON ITAB(300),ISEC(300),IFIRST,M,F,DJ(600),                          VIR0003
     1DUM2(1800),DUM3(365100),DUM4(4057),IDUM2(905),
     2IDUM5(1203),IX11,IX12,IX13,IX14,IX15,IDUM6(2),DUM7(130200),
     1NLS,LN,IFLAG,IERP,ITRANS,ISTED
      REAL MEA1,MEA2,MEA3,MEA5,MEA6,MEAU1,MEAU2,MEAU3,MEAU5,MEAU6             VIR0004
C
C   CALCULATE VIRTUAL WORK CONTRIBUTION TO DISPLACEMENT CONSTRAINT DIRECT
C                                                                   ION
C   OF MEMBER I
C
      NPL=IGPT(LL,2)+2                                                         VIR0005
      XMIN=10000.0                                                             VIR0006
      ITABI=ITAB(I)                                                            VIR0007
      ISECI=ISEC(I)                                                            VIR0008
      READ(ITABI"ISECI) AREA,RIX,RIY,RIZ                                       VIR0009
C
C   READ IN MEAS FOR UNIT LOAD IN QUESTION
C
      IFILE=20+IGPT(LL,1)                                                      VIR0010
      READ(IFILE"I) MEAU1,MEAU2,MEAU3,MEAU5,MEAU6                              VIR0011
      READ(19"I) RHO,U,XL                                                      VIR0012
      DO 640 INP=3,NPL                                                         VIR0013
      J=IGPT(LL,INP)                                                           VIR0014
C
C   READ IN MEAS FOR REAL LOAD IN QUESTION
C
      JFILE=20+J                                                               VIR0015
      READ(JFILE"I) MEA1,MEA2,MEA3,MEA5,MEA6                                   VIR0016
      YA=(MEA5-MEA2)/XL                                                        VIR0017
      YC=(MEAU5-MEAU2)/XL                                                      VIR0018
      ZA=(MEA6-MEA3)/XL                                                        VIR0019
      ZC=(MEAU6-MEAU3)/XL                                                      VIR0020
C
C   CONTRIBUTION OF Y AXIS BENDING
C
      BY1=YA*YC*(XL**3)/3.0                                                    VIR0021
      BY2=((MEA2*YC)+(YA*MEAU2))*(XL**2)/2.0                                   VIR0022
      BY3=MEA2*MEAU2*XL                                                        VIR0023
      BY=BY1+BY2+BY3                                                           VIR0024
```

```
C
C  CONTRIBUTION OF Z AXIS BENDING
C
       BZ1=ZA*ZC*(XL**3)/3.0
       BZ2=((MEA3*ZC)+(ZA*MEAU3))*(XL**2)/2.0
       BZ3=MEA3*MEAU3*XL
       BZ=BZ1+BZ2+BZ3
C
C  CONTRIBUTION OF AXIAL EFFECTS
C
       AXIAL=MEA1*MEAU1*XL
       IF(IDERIV.EQ.1) GO TO 630
       IF(IN.EQ.1) GO TO 620
       DELVW(LL,J)=((AXIAL/AREA)+(BY/RIY)+(BZ/RIZ))/E
       GO TO 640
  620  DELVWN(LL,J)=((AXIAL/AREA)+(BY/RIY)+(BZ/RIZ))/E
       GO TO 640
  630  NSEC=ISEC(I)+1
       ITABI=ITAB(I)
       READ(ITABI'NSEC) AREAN,RIXN,RIYN,RIZN
       C1=AXIAL/(AREA*AREA)
       C2=1.0/(AREAN-AREA)
       C3=(RIZN-RIZ)/(RIZ*RIZ)
       C4=(RIYN-RIY)/(RIY*RIY)
       DDDA=-(C1+C2*((C3*BZ)+(C4*BY)))/E
       QQ=DDDA/(RHO*U*XL)
       IF((XMIN-QQ).LT..0001) GO TO 640
       XMIN=QQ
       Q(I,LL)=QQ
  640  CONTINUE
       RETURN
       END
```

```
*DECK COSTWT                                                                COS0000
      SUBROUTINE COSTWT                                                     COS0001
      COMMON Q(300,3),DELEX(3,6),DERROR(3,6),DVW(3,6),DELVWN(3,6),         COS0002
     1DELVW(3,6),XJT(3,4),DC(3),QDMIN(3),THETA(2),FACT,IREG,JCHK,
     2IQPT(3,6),JT(3,2),IDTYP(3),ICK(3),IMEM(3),MOD(3),ICHG(3),IDC(3),
     3ICOL,IBEAM,IBR,NDC,ITER,IDERIV,MEMCHG,IN,JMEM,LL,I,IX16,IX17,IX18,
     4IX19,IX20,IX21,IX22,IY23,IX24,IX25,IY26,IY27,IMAX(300),NC,NP,NBR,
     4NCS,NBS,NBRS,
     5COLID1(48),COLID2(48),BMID1(38),BMID2(38),BRID1(20),BRID2(20)
      COMMON ITAB(300),ISEC(300),IFIRST,M,F,DJ(600),                        COS0003
     1DUM2(1800),DUM3(365100),DUM4(4057),IDUM2(905),
     2IDUM5(1203),IY11,IX12,IX13,IX14,IX15,IDUM6(2),DUM7(130200),
     1NLS,LN,IFLAG,IERR,ITRANS,ISTEP
C
C  CALCULATE WEIGHT AND COST OF STRUCTURE
C
      CW=0.0                                                                COS0004
      CC=0.0                                                                COS0005
      BW=0.0                                                                COS0006
      BC=0.0                                                                COS0007
      BRW=0.0                                                               COS0008
      BRC=0.0                                                               COS0009
      DO 50 I=1,M                                                           COS0010
      J=ISEC(I)                                                             COS0011
      READ(19"I) RHO,U,XL                                                   COS0012
      IF(ITAB(I).EQ.16) GO TO 10                                            COS0013
      IF(ITAB(I).EQ.17) GO TO 20                                            COS0014
      READ(18"J) AREA,RIX,RIY,RIZ                                           COS0015
      ADW=(AREA*XL*RHO)/1728.0                                              COS0016
      BRW=BRW+ADW                                                           COS0017
      BRC=BRC+(ADW*U)                                                       COS0018
      GO TO 50                                                              COS0019
   10 READ(16"J) AREA,RIX,RIY,RIZ                                           COS0020
      ADW=(AREA*XL*RHO)/1728.0                                              COS0021
      CW=CW+ADW                                                             COS0022
      CC=CC+(ADW*U)                                                         COS0023
      GO TO 50                                                              COS0024
   20 READ(17"J) AREA,RIX,RIY,RIZ                                           COS0025
      ADW=(AREA*XL*RHO)/1728.0                                              COS0026
      BW=BW+ADW                                                             COS0027
      BC=BC+(ADW*U)                                                         COS0028
   50 CONTINUE                                                              COS0029
      CW=CW/2000.0                                                          COS0030
      BW=BW/2000.0                                                          COS0031
      BRW=BRW/2000.0                                                        COS0032
      TW=CW+BW+BRW                                                          COS0033
      TC=CC+BC+BRC                                                          COS0034
      WRITE(6,60) CW,CC                                                     COS0035
```

```
60 FORMAT(1X,"TOTAL COLUMN WEIGHT IN TONS IS",F12.2,2X,                  COS0036
  1"WHICH COSTS $",F12.2/)
   WRITE(6,65) BW,BC                                                    COS0037
65 FORMAT(1X,"TOTAL BEAM   WEIGHT IN TONS IS",F12.2,2X,                  COS0038
  1"WHICH COSTS $",F12.2/)
   WRITE(6,70) BRW,BRC                                                  COS0039
70 FORMAT(1X,"TOTAL BRACE  WEIGHT IN TONS IS",F12.2,2X,                  COS0040
  1"WHICH COSTS $",F12.2/)
   WRITE(6,75) TW,TC                                                    COS0041
75 FORMAT(1X,"TOTAL BLDG.  WEIGHT IN TONS IS",F12.2,2X,                  COS0042
  1"WHICH COSTS $",F12.2//)
   RETURN                                                               COS0043
   END                                                                  COS0044
```

```
*DECK OUTPUT                                                                        OUT0000
      SUBROUTINE OUTPUT                                                             OUT0001
      COMMON Q(300,3),DELEX(3,6),DERROR(3,6),DVW(3,6),DELVWN(3,6),                 OUT0002
     1DELVW(3,6),XJT(3,4),DC(3),ODMIN(3),THETA(2),FACT,IRFG,JCHK,
     2IOPT(3,6),JT(3,2),IDTYP(3),ICK(3),IMEM(3),MOD(3),ICHG(3),IDC(3),
     3ICOL,IBEAM,IBR,NDC,ITER,IDERIV,MEMCRG,IN,JMEM,LL,I,IX16,IX17,IX18,
     4IX19,IX20,IX21,IX22,IX23,IX24,IX25,IX26,IX27,IMAX(300),NC,NB,NBR,
     4NCS,NBS,NBRS,
     5COLID1(48),COLID2(48),BMID1(38),BMID2(38),BRID1(20),BRID2(20)
      COMMON ITAB(300),ISEC(300),IFIRST,M,E,DJ(600),                               OUT0003
     1DUM2(1800),DUM3(365100),DUM4(4057),IDUM2(905),
     2IDUM5(1203),IX11,IX12,IX13,IX14,IX15,IDUM6(2),DUM7(130200),
     1NLS,LN,IFLAG,IERR,ITRANS,ISTED
      IF(JCHK.EQ.0) GO TO 100                                                      OUT0004
      WRITE(6,410)                                                                 OUT0005
  410 FORMAT(1H1,"**********************************************")                OUT0006
      WRITE(6,415)                                                                 OUT0007
  415 FORMAT(1X,"FINAL MEMBER PROPERTIES -- COMPLETE DESIGN")                      OUT0008
      WRITE(6,420)                                                                 OUT0009
  420 FORMAT(1X,"**********************************************"//)              OUT0010
      GO TO 80                                                                     OUT0011
  100 WRITE(6,63)                                                                  OUT0012
   63 FORMAT(1H1,"*************************")                                     OUT0013
      WRITE(6,64)                                                                  OUT0014
   64 FORMAT(1X,"INITIAL MEMBER PROPERTIES")                                       OUT0015
      WRITE(6,66)                                                                  OUT0016
   66 FORMAT(1X,"*************************"//)                                   OUT0017
      GO TO 80                                                                     OUT0018
   80 WRITE(6,75)                                                                  OUT0019
   75 FORMAT(1X,"NUMBER      NAME      AX         IX          IY          IZ       OUT0020
     1RHO      U          L")
      DO 50 I=1,M                                                                  OUT0021
      J=ISEC(I)                                                                    OUT0022
      READ(19"I) RHO,U,XL                                                          OUT0023
      IF(ITAB(I).EQ.16) GO TO 10                                                   OUT0024
      IF(ITAB(I).EQ.17) GO TO 20                                                   OUT0025
      READ(18"J) AREA,RIX,RIY,RIZ                                                  OUT0026
      WRITE(6,15) I,BRID1(J),BRID2(J),AREA,RIX,RIY,RIZ,RHO,U,XL                    OUT0027
   15 FORMAT(1X,I4,3X,2A4,F8.2,4F9.2,F7.2,F9.2)                                   OUT0028
      GO TO 50                                                                     OUT0029
   10 READ(16"J) AREA,RIX,RIY,RIZ                                                  OUT0030
      WRITE(6,15) I,COLID1(J),COLID2(J),AREA,RIX,RIY,RIZ,RHO,U,XL                  OUT0031
      GO TO 50                                                                     OUT0032
   20 READ(17"J) AREA,RIX,RIY,RIZ                                                  OUT0033
      WRITE(6,15) I,BMID1(J),BMID2(J),AREA,RIX,RIY,RIZ,RHO,U,XL                    OUT0034
      GO TO 50                                                                     OUT0035
   50 CONTINUE                                                                     OUT0036
      IF(JCHK.EQ.0) GO TO 8                                                        OUT0037
```

```
      WRITE(6,55)
   55 FORMAT(1H1,"*****************")
      WRITE(6,60)
   60 FORMAT(1X,"FINAL DESIGN DATA")
      WRITE(6,65)
   65 FORMAT(1X,"*****************"//)
      WRITE(6,178) NCS
  178 FORMAT(1X,"NUMBER OF COLUMNS CHANGED -----------",I5/)
      WRITE(6,179) NBS
  179 FORMAT(1X,"NUMBER OF BEAMS   CHANGED -----------",I5/)
      WRITE(6,181) NBRS
  181 FORMAT(1X,"NUMBER OF BRACES  CHANGED -----------",I5/)
      WRITE(6,182) ITER
  182 FORMAT(1X,"TOTAL NUMBER OF MEMBERS CHANGED -----",I5//)
      WRITE(6,4) IREG
    4 FORMAT(1X,"NUMBER OF REGULAR ANALYSES ----------",I5/)
      WRITE(6,5) ISTED
    5 FORMAT(1X,"NUMBER OF STEDES ANALYSES -----------",I5/)
      WRITE(6,6) JCHK
    6 FORMAT(1X,"TOTAL NUMBER OF ANALYSES ------------",I5//)
      WRITE(6,699)
  699 FORMAT(1X,"*******************************************")
      WRITE(6,700)
  700 FORMAT(1X,"FINAL MATERIAL WEIGHT AND COST EVALUATION")
      WRITE(6,701)
  701 FORMAT(1X,"********************************************"//)
      CALL COSTWT
    8 RETURN
      END
```

```
*DECK STFDES                                                                    STF0000
      SUBROUTINE STFDES                                                         STF0001
      COMMON DUM5(972),DUM6(39),IDUM3(44),IDUM4(14),IDUM5(315),DUM8(212)STF0002
      COMMON IDUM1(602),DUM1(601),DUM2(1800),DUM3(365100),DUM4(4057),           STF0003
     1IDUM2(905),
     5RL(600),CRL(600),N,UBW,NJ,IX11,IX12,IX13,IX14,IX15,NROW,NUBW,
     6ST(360,360),D(600),
     1NLS,LN,IFLAG,IERR,ITRANS,ISTFD
      COMMON IDEP(100,8),NJP,IR(100),JROW(100,7),NRJT,IOR(600),JD,              STF0004
     1INDEP,IDIAG(100),IRR,INR,INC,KTREC,IPART,IJK,KLM,ICROW,II,JJ,
     2S(6,600),LLD,STEMP(6,600),DUM(6,6),KUBW,NF,NFUL1,IP,T(600,6),
     3D1(600),IFUL1,JLIM,NF,XP,ZP,JTN(16),MJ(2,2),JF(24),ISJ(24,32)
      INTEGER UBW,RL,CRL                                                        STF0005
C
C  INPUT STFDES PERTINENT INFORMATION AND PERFORM DATA CONSISTENCY CHECK
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                               CCCCCCCC
C
      IF(ISTFD.NE.0) GO TO 471                                                  STF0006
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                               CCCCCCCC
      CALL CONCHK                                                               STF0007
      IF(IERR.EQ.1) GO TO 1                                                     STF0008
C
C  CONVERT LOCAL DISPLACEMENT COMPONENT NUMBERS TO GLOBAL
C
      CALL GLOBAL                                                               STF0009
C
C  GENERATE REORDERING VECTOR IOR
C
      CALL ORDER                                                                STF0010
C
C  IDIAG(I)=RECORD NUMBER OF THE ITH DIAGONAL POSITION IN REORDERED K.
C
      IDIAG(1)=1                                                                STF0011
      ISUM=0                                                                    STF0012
      NRJT1=NRJT-1                                                              STF0013
      DO 470 I1=1,NRJT1                                                         STF0014
      I2=I1+1                                                                   STF0015
      I=NRJT+1-I1                                                               STF0016
      ISUM=ISUM+I                                                               STF0017
      IDIAG(I2)=ISUM+1                                                          STF0018
  470 CONTINUE                                                                  STF0019
      CALL FORMT                                                                STF0020
      JLIM=INDEP+IRR                                                            STF0021
  471 INR=0                                                                     STF0022
```

```
      INC=0
      KTREC=0
      IPART=0
      IJK=0
      KLM=0
      ICROW=0
      CALL UNPACK
C
C  ZERO OUT ARRAY USED TO STORE FINAL REDUCED STIFFNESS MATRIX
C
      DO 5000 I=1,360
      DO 5000 J=1,360
 5000 ST(I,J)=0.0
C
C  PERFORM OPERATIONS ON PARTITIONED ROW II OF STIFFNESS MATRIX
C
      INR=0
      DO 990 II=1,NRJT
C
C  ZERO OUT ARRAY USED TO STORE PARTITIONED ROW II
C
      DO 34 I=1,6
      DO 34 J=1,600
   34 STEMP(I,J)=0.0
      IK=0
      INR1=INR+1
      INR=INR+IR(II)
C
C  READ IN FROM STORAGE THE ROWS OF THE STIFFNESS MATRIX IN THE NEW
C  ORDER DETERMINED BY DEPENDENT AND INDEPENDENT DISPLACEMENTS
C
      DO 2 I=INR1,INR
      IK=IK+1
      DO 3 KM=1,N
      IF(IOR(KM).EQ.I) GO TO 4
    3 CONTINUE
    4 K=UBW+KM-1
      IF(K.GT.N) K=N
      READ(11'KM)(STEMP(IK,J),J=1,K)
    2 CONTINUE
C
C  DETERMINE IF PARTITICNED ROW II IS COMPLFTELY ABOVE HORIZONTAL
C  PARTITION OF K(PARTITION OF K11 AND K12) (IPART=0), OR STRADDLING
C  HORIZONTAL PARTITION(IJK IS GREATER THAN ZERO), OR COMPLFTELY BELOW
C  HORIZONTAL PARTITION(IPART=1)
C
      IF((INR-IR(II)+1).GT.INDEP) GO TO 670
      IRII=IR(II)
```

```
      DO 660 KKK=1,IRII                                                STF0052
      IF((INR-KKK+1).LE.INDEP) GO TO 690                               STF0053
  660 CONTINUE                                                         STF0054
      GO TO 700                                                        STF0055
C
C  PARTITIONED ROW II IS ENTIRELY BELOW HORIZONTAL PARTITION OF K
C
  670 IPART=1                                                          STF0056
C
C  KLM=FIRST PARTITIONED ROW NUMBER WHICH IS COMPLETELY BELOW HORIZONTAL
C  PARTITION IN K
C
      IF(KLM.LE.0) GO TO 680                                           STF0057
      GO TO 700                                                        STF0058
  680 KLM=II                                                           STF0059
      GO TO 700                                                        STF0060
C
C  IJK=NUMBER OF ROWS IN PARTITIONED ROW II BELOW THE HORIZONTAL PARTITI
C                                                          ON OF K
C
  690 IJK=KKK-1                                                        STF0061
      GO TO 700                                                        STF0062
  700 CALL ROWBND                                                      STF0063
      IF(IERR.EQ.1) GO TO 1                                            STF0064
      CALL RPKKT                                                       STF0065
      CALL STORE                                                       STF0066
  990 CONTINUE                                                         STF0067
      CALL TTRKT                                                       STF0068
    1 RETURN                                                           STF0069
      END                                                              STF0070
```

```
*DECK CONCHK
      SUBROUTINE CONCHK
      COMMON DUM5(972),DUM6(39),IDUM3(44),IDUM4(14),IDUM5(315),DUM8(212)
      COMMON IDUM1(602),DUM1(601),DUM2(1800),DUM3(365100),DUM4(4057),
     1IDUM2(905),
     5RL(600),CRL(600),N,URW,NJ,IX11,IX12,IX13,IX14,IX15,NROW,NUBW,
     6ST(360,360),D(600),
     1NLS,LN,IFLAG,IEPR,ITRANS,ISTEP
      COMMON IDEP(100,8),NJD,IR(100),JROW(100,7),NPJT,ICR(600),JD,
     1INDEP,IDIAG(100),IPR,INR,INC,KTREC,IPART,IJK,KLM,ICROW,II,JJ,
     2S(6,600),LLD,STEMP(6,600),DUM(6,6),KUPW,NP,NFUL1,IP,T(600,6),
     3D1(600),IFUL1,JLIM,NF,XP,ZP,JTN(16),MJ(2,2),JF(24),ISJ(24,32)
      INTEGER URW,RL,CRL
      EXTERNAL KFN
      IF(ITRANS.EQ.0) GO TO 7
C
C   PERFORM CONSISTENCY CHECK ON DATA INPUT FOR INTERNAL GENERATION OF T
C                                                                 MATRIX
C
C
C   INPUT NO. OF FLOORS,JOINTS ON EACH FLOOR AND X AND Z COORDINATES OF
C   CENTER OF RIGID BODY
C
      IEPR=0
      READ(5,5) NF,XP,ZP
    5 FORMAT(I5,2F10.2)
      IF(NF.LE.0) GO TO 6
      GO TO 9
    6 IEPR=1
      WRITE(6,8) NF
    8 FORMAT(1X,"NO.OF FLOORS=",I4,2X,"WHICH IS IMPOSSIBLE"/)
      GO TO 270
    9 DO 205 I=1,NF
      READ(5,15) IFN,JF(IFN),NCF
   15 FORMAT(3I5)
      IC=0
      DO 195 J=1,NCF
      READ(5,25) (JTN(J1),J1=1,16)
   25 FORMAT(16I5)
      DO 185 K=1,16
      IF(JTN(K).EQ.0) GO TO 35
      IF(JTN(K).LT.0) GO TO 105
      IC=IC+1
      ISJ(IFN,IC)=JTN(K)
      GO TO 185
   35 IF(IC.EQ.0) GO TO 85
      IF(IC.EQ.JF(IFN)) GO TO 195
      IEPR=1
```

```
      IF(J.EQ.NCF) GO TO 65
      WRITE(6,45) IFN
   45 FORMAT(1X,"DATA INPUT ERROR FOR FLOOR NO.",I5/)
      WRITE(6,55)
   55 FORMAT(1X,"ZERO SPECIFIED AS A JOINT"/)
      WRITE(6,551)
  551 FORMAT(1X,"INFORMATION ON DATA CARD"/)
      WRITE(6,552) (JTN(J2),J2=1,16)
  552 FORMAT(1X,I4,15I5/)
      GO TO 185
   65 WRITE(6,45) IFN
      WRITE(6,75)
   75 FORMAT(1X,"NO. OF JOINTS ON CARDS DOES NOT EQUAL NUMBER READ IN"/)
      WRITE(6,551)
      WRITE(6,552) (JTN(J2),J2=1,16)
      GO TO 195
   85 IERR=1
      WRITE(6,45) IFN
      WRITE(6,95)
   95 FORMAT(1X,"FIRST JOINT ON FLOOR SPECIFIED AS ZERO"/)
      WRITE(6,551)
      WRITE(6,552) (JTN(J2),J2=1,16)
      GO TO 185
  105 IF(K.EQ.1) GO TO 125
      IF(JTN(K-1).LT.0) GO TO 145
      JTNK=JTN(K)
      JTNK1=JTN(K-1)
      IF(IABS(JTNK).LE.JTNK1) GO TO 155
      JTNK1=JTNK1+1
      JTNK2=IABS(JTNK)
      DO 115 K1=JTNK1,JTNK2
      IC=IC+1
      ISJ(IFN,IC)=K1
  115 CONTINUE
      GO TO 185
  125 WRITE(6,45) IFN
      WRITE(6,135)
  135 FORMAT(1X,"FIRST JOINT SPECIFIED AS NEGATIVE"/)
      WRITE(6,551)
      WRITE(6,552) (JTN(J2),J2=1,16)
      IERR=1
      GO TO 185
  145 WRITE(6,45) IFN
      WRITE(6,155)
  155 FORMAT(1X,"TWO CONSECUTIVE JOINTS SPECIFIED AS NEGATIVE"/)
      WRITE(6,551)
      WRITE(6,552) (JTN(J2),J2=1,16)
      IERR=1
```

```
      GO TO 185
  165 WRITE(6,45) IFN
      WRITE(6,175)
  175 FORMAT(1X,"CONSECUTIVE JOINTS NOT IN ORDER"/)
      WRITE(6,551)
      WRITE(6,552) (JTN(J2),J2=1,16)
      IERR=1
      GO TO 185
  185 CONTINUE
  195 CONTINUE
  205 CONTINUE
      IF(IERR.EQ.1) GO TO 270
C
C FORM IDEP ARRAY IN CORE
C
      NJD=0
      DO 225 I=1,NF
      JFI=JF(I)
      DO 215 J=1,JFI
      NJD=NJD+1
      IDEP(NJD,1)=IFJ(I,J)
      IF(ITRANS.EQ.1) GO TO 213
      IDEP(NJD,2)=3
      IDEP(NJD,3)=1
      IDEP(NJD,4)=3
      IDEP(NJD,5)=5
      IDEP(NJD,6)=0
      IDEP(NJD,7)=0
      IDEP(NJD,8)=0
      GO TO 215
  213 IDEP(NJD,2)=6
      IDEP(NJD,3)=1
      IDEP(NJD,4)=2
      IDEP(NJD,5)=3
      IDEP(NJD,6)=4
      IDEP(NJD,7)=5
      IDEP(NJD,8)=6
      GO TO 215
  215 CONTINUE
  225 CONTINUE
      JD=3*NJD
      IF(ITRANS.EQ.1) JD=6*NJD
C
C MAKE SURE NO JOINT IS SPECIFIED TWICE
C
      KJT=0
      JTMIN=1
  500 IDUP=0
```

```
      DO 530 I=1,NF
      JFI=JF(I)
      DO 520 J=1,JFI
      IF(ISJ(I,J).EQ.JTMIN) GO TO 510
      GO TO 520
  510 IDUP=IDUP+1
      KJT=KJT+1
      GO TO 520
  520 CONTINUE
  530 CONTINUE
      IF(IDUP.GT.1) GO TO 540
  535 IF(KJT.EQ.NJD) GO TO 270
      JTMIN=JTMIN+1
      GO TO 500
  540 IERR=1
      WRITE(6,550) JTMIN,IDUP
  550 FORMAT(1X,"INPUT ERROR--JOINT",I4,2X,"APPEARS",I4,2X,"TIMES"/)
      GO TO 535
C
C    INPUT NJD,JD AND IDEP ARRAY
C
    7 IERR=0
      READ(5,10) NJD,JD
   10 FORMAT(2I5)
C
C    CHECK TO SEE IF NJD JOINTS IS GREATER THAN ZERO AND CHECK THAT
C    THE NO. OF DEPENDENT DISPLACEMENTS IS GREATER THAN OR EQUAL TO
C    NJD JOINTS
C
      IF(NJD.GT.0.AND.JD.GE.NJD) GO TO 40
      WRITE(6,20)
   20 FORMAT(1X,"NJD OR JD HAS BEEN INCORRECTLY SPECIFIED")
      WRITE(6,30) NJD,JD
   30 FORMAT(1X,"NJD=",I5,"JD=",I5)
      GO TO 270
   40 DO 52 J=1,NJD
      READ(5,50) (IDEP(J,I),I=1,8)
   50 FORMAT(8I5)
   52 CONTINUE
C
C    MAKE SURE NO JOINT IS SPECIFIED TWICE
C
      KJT=0
      JTMIN=1
  600 IDUP=0
      DO 620 I=1,NJD
      IF(IDEP(I,1).EQ.JTMIN) GO TO 610
      GO TO 620
```

```
  610 IDUP=IDUP+1
      KJT=KJT+1
      GO TO 620
  620 CONTINUE
      IF(IDUP.GT.1) GO TO 630
  625 IF(KJT.EQ.NJD) GO TO 635
      JTMIN=JTMIN+1
      GO TO 600
  630 IERR=1
      WRITE(6,550) JTMIN,IDUP
      GO TO 625
  635 IF(IERR.EQ.1) GO TO 270
C
C  CHECK THAT THE NO. OF DEPENDENT DISPLACEMENTS INPUT IS GREATER THAN Z
C                                                               ERO
C
      DO 230 J=1,NJD
      IF(IDEP(J,2).GT.0) GO TO 80
      IERR=1
      WRITE(6,60)
   60 FORMAT(1X,"NEGATIVE OR ZERO NO. OF DEPENDENT DISPLACEMENTS SPECIFI
     1ED")
      WRITE(6,70) IDEP(J,2)
   70 FORMAT(1X,"IDEP(J,2)=",I5)
   80 LIM=2+IDEP(J,2)
C
C  CHECK THAT ONLY COMPONENT DIRECTIONS 1 TO 6 ARE SPECIFIED
C
      DO 110 I=3,LIM
      IF(IDEP(J,I).LE.6.AND.IDEP(J,I).GE.1) GO TO 110
      IERR=1
      WRITE(6,90)
   90 FORMAT(1X,"LOCAL DEPENDENT DISPLACEMENT COMPONENT IMPROPERLY SPECI
     1FIED AS A NO. LESS THAN 1 OR GREATER THAN 6")
      WRITE(6,100) (IDEP(J,IJ),IJ=1,8)
  100 FORMAT(1X,"JOINT INFORMATION=",8I5)
  110 CONTINUE
C
C  VERIFY THAT THE DEPENDENT DISPLACEMENT NUMBERS AT EACH OF NJD JOINTS
C  IS IN ASCENDING ORDER
C
      IF(IDEP(J,2).EQ.1) GO TO 230
      IF(RL(KFN(J,3)).NE.1.OR.RL(KFN(J,4)).NE.1) GO TO 130
      IERR=1
      WRITE(6,120)
  120 FORMAT(1X,"A DEPENDENT DISPLACEMENT COMPONENT HAS BEEN SPECIFIED A
     1S RESTRAINED")
      WRITE(6,100) (IDEP(J,IJ),IJ=1,8)
```

```
  130 IF(IDEP(J,3).LT.IDEP(J,4)) GO TO 150
      IERR=1
      WRITE(6,140)
  140 FORMAT(1X,"DEPENDENT DISPLACEMENT COMPONENTS NOT IN ASCENDING ORDE
     1R")
      WRITE(6,100) (IDEP(J,IJ),IJ=1,8)
  150 IF(IDEP(J,2).LE.2) GO TO 230
      IF(RL(KFN(J,5)).NE.1) GO TO 160
      IERR=1
      WRITE(6,120)
      WRITE(6,100) (IDEP(J,IJ),IJ=1,8)
  160 IF(IDEP(J,4).LT.IDEP(J,5)) GO TO 170
      IERR=1
      WRITE(6,140)
      WRITE(6,100) (IDEP(J,IJ),IJ=1,8)
  170 IF(IDEP(J,2).LE.3) GO TO 230
      IF(RL(KFN(J,6)).NE.1) GO TO 180
      IERR=1
      WRITE(6,120)
      WRITE(6,100) (IDEP(J,IJ),IJ=1,8)
  180 IF(IDEP(J,5).LT.IDEP(J,6)) GO TO 190
      IERR=1
      WRITE(6,140)
      WRITE(6,100) (IDEP(J,IJ),IJ=1,8)
  190 IF(IDEP(J,2).LE.4) GO TO 230
      IF(RL(KFN(J,7)).NE.1) GO TO 200
      IERR=1
      WRITE(6,120)
      WRITE(6,100) (IDEP(J,IJ),IJ=1,8)
  200 IF(IDEP(J,6).LT.IDEP(J,7)) GO TO 210
      IERR=1
      WRITE(6,140)
      WRITE(6,100) (IDEP(J,IJ),IJ=1,8)
  210 IF(IDEP(J,2).LE.5) GO TO 230
      IF(RL(KFN(J,8)).NE.1) GO TO 220
      IERR=1
      WRITE(6,120)
      WRITE(6,100) (IDEP(J,IJ),IJ=1,8)
  220 IF(IDEP(J,7).LT.IDEP(J,8)) GO TO 230
      IERR=1
      WRITE(6,140)
      WRITE(6,100) (IDEP(J,IJ),IJ=1,8)
  230 CONTINUE
      IF(IERR.EQ.0) GO TO 240
      GO TO 270
  240 JDTOT=0
C
C     CHECK THAT THE NO. OF DEPENDENT DISPLACEMENT COMPONENTS JD EQUAL THE
```

```
C  THE NUMBER READ IN
C
      DO 250 J=1,NJD
      JDTOT=JDTOT+IDEP(J,2)
  250 CONTINUE
      IF(JDTOT.EQ.JD) GO TO 270
      WRITE(6,260)
  260 FORMAT(1X,"NO. OF DEPENDENT DISPLACEMENT COMPONENTS ADDED UP FROM
     1EACH OF NJD JOINTS IS NOT EQUAL TO JD READ IN")
      WRITE(6,30) NJD,JD
      IERR=1
  270 RETURN
      END
```

```
*DECK GLOBAL                                                               GLO00000
      SUBROUTINE GLOBAL                                                    GLO00001
      COMMON DUM5(972),DUM6(39),IDUM3(44),IDUM4(14),IDUM5(315),DUM8(212)GLO00002
      COMMON IDUM1(602),DUM1(601),DUM2(1800),DUM3(365100),DUM4(4057),    GLO00003
     1IDUM2(905),
     5RL(600),CRL(600),N,UBW,NJ,IX11,IX12,IX13,IX14,IX15,NROW,NUBW,
     6ST(360,360),D(600),
     1NLS,LN,IFLAG,IERR,ITRANS,ISTEP
      COMMON IDEP(100,8),NJD,IR(100),JROW(100,7),NRJT,ICR(600),JD,       GLO00004
     1INDEP,IDIAG(100),IRP,INR,INC,KTREC,IPART,IJK,KLM,ICROW,II,JJ,
     2S(6,600),LLD,STEMP(6,600),DUM(6,6),KUPW,NP,NFUL1,IP,T(600,6),
     3D1(600),IFUL1,JLIM,NF,XP,ZF,JTN(16),MJ(2,2),JF(24),ISJ(24,32)
      INTEGER UBW,RL,CRL                                                   GLO00005
      INTEGER DD1,DD2,DD3,DD4,DD5,DD6                                      GLO00006
C
C   CONVERT LOCAL DEPENDENT DISPLACEMENT COMPONENT NUMBERS (1 TO 6)
C   TO GLOBAL UNKNOWN COMPONENT NUMBERS (1 TO (6*NJ-CRL(6*NJ)))
C
      DO 330 J=1,NJD                                                       GLO00007
      DD1=6*IDEP(J,1)-5                                                    GLO00008
      DD2=DD1+1                                                            GLO00009
      DD3=DD2+1                                                            GLO00010
      DD4=DD3+1                                                            GLO00011
      DD5=DD4+1                                                            GLO00012
      DD6=DD5+1                                                            GLO00013
      KLIM=2+IDEP(J,2)                                                     GLO00014
C
C   IDEP(J,I)=ORDER OF THE ITH DEPENDENT DISPLACEMENT UNKNOWN IN A
C   VECTOR OF DISPLACEMENT UNKNOWNS IN NATURAL ORDER. (FOR I=3,4...IDEP(J
C                                                                 ,2))
C
      DO 330 I=3,KLIM                                                      GLO00015
      IF(IDEP(J,I).NE.1) GO TO 280                                         GLO00016
      IDEP(J,I)=DD1-CRL(DD1)                                               GLO00017
      GO TO 330                                                            GLO00018
  280 IF(IDEP(J,I).NE.2) GO TO 290                                         GLO00019
      IDEP(J,I)=DD2-CRL(DD2)                                               GLO00020
      GO TO 330                                                            GLO00021
  290 IF(IDEP(J,I).NE.3) GO TO 300                                         GLO00022
      IDEP(J,I)=DD3-CRL(DD3)                                               GLO00023
      GO TO 330                                                            GLO00024
  300 IF(IDEP(J,I).NE.4) GO TO 310                                         GLO00025
      IDEP(J,I)=DD4-CRL(DD4)                                               GLO00026
      GO TO 330                                                            GLO00027
  310 IF(IDEP(J,I).NE.5) GO TO 320                                         GLO00028
      IDEP(J,I)=DD5-CRL(DD5)                                               GLO00029
      GO TO 330                                                            GLO00030
  320 IF(IDEP(J,I).NE.6) GO TO 330                                         GLO00031
```

```
      IDEP(J,I)=DD6-CRL(DD6)                                             GLO00032
      GO TO 330                                                          GLO00033
  330 CONTINUE                                                           GLO00034
      DO 340 J=1,7                                                       GLO00035
      DO 340 I=1,500                                                     GLO00036
      IR(I)=0                                                            GLO00037
      JROW(I,J)=0                                                        GLO00038
  340 CONTINUE                                                           GLO00039
C
C    JROW(J,I)=NUMBER OF RELEASED DISPLACEMENT COMPONENTS AT JOINT J AND T
C                                                                       HE
C    ORDER OF THE ITH DISPLACEMENT AT JOINT J IN A VECTOR OF DISPLACEMENT
C    UNKNOWNS IN NATURAL ORDER.  IF RESTRAINED IT EQUALS 0.
C    SET UP ARRAYS TO STORE NUMBER OF RELEASED DISPLACEMENT COMPONENTS
C    PER JOINT AND NUMBER OF ABSOLUTE ROWS PER PARTITIONED ROW.
C    NRJT=NUMBER OF PARTITIONED ROWS CONTAINING UNKNOWN DISPLACEMENT COMPO
C                                                                    NENTS
C
      NRJT=0                                                             GLO00040
      DO 410 J=1,NJ                                                      GLO00041
      DD1=6*J-5                                                          GLO00042
      DD2=DD1+1                                                          GLO00043
      DD3=DD2+1                                                          GLO00044
      DD4=DD3+1                                                          GLO00045
      DD5=DD4+1                                                          GLO00046
      DD6=DD5+1                                                          GLO00047
      IF(RL(DD1).EQ.1.AND.RL(DD2).EQ.1.AND.RL(DD3).EQ.1.AND.RL(DD4).EQ.1GLO00048
     1.AND.RL(DD5).EQ.1.AND.RL(DD6).EQ.1) GO TO 410
      NRJT=NRJT+1                                                        GLO00049
      I=1                                                                GLO00050
      IF(RL(DD1).NE.0) GO TO 350                                         GLO00051
      I=I+1                                                              GLO00052
      JROW(J,I)=DD1-CRL(DD1)                                             GLO00053
  350 IF(RL(DD2).NE.0) GO TO 360                                         GLO00054
      I=I+1                                                              GLO00055
      JROW(J,I)=DD2-CRL(DD2)                                             GLO00056
  360 IF(RL(DD3).NE.0) GO TO 370                                         GLO00057
      I=I+1                                                              GLO00058
      JROW(J,I)=DD3-CRL(DD3)                                             GLO00059
  370 IF(RL(DD4).NE.0) GO TO 380                                         GLO00060
      I=I+1                                                              GLO00061
      JROW(J,I)=DD4-CRL(DD4)                                             GLO00062
  380 IF(RL(DD5).NE.0) GO TO 390                                         GLO00063
      I=I+1                                                              GLO00064
      JROW(J,I)=DD5-CRL(DD5)                                             GLO00065
  390 IF(RL(DD6).NE.0) GO TO 400                                         GLO00066
      I=I+1                                                              GLO00067
      JROW(J,I)=DD6-CRL(DD6)                                             GLO00068
```

```
  400 JROW(J,I)=I-1                                    GLO00065
      IR(NRJT)=I-1                                     GLO00070
  410 CONTINUE                                         GLO00071
      RETURN                                           GLO00072
      END                                              GLO00073
```

```
*DECK ORDER                                                             ORD0000
      SUBROUTINE ORDER                                                  ORD0001
      COMMON DUM5(972),DUM6(39),IDUM3(44),IDUM4(14),IDUM5(315),DUM8(212)ORD0002
      COMMON IDUM1(602),DUM1(601),DUM2(1800),DUM3(365100),DUM4(4057),   ORD0003
     1IDUM2(905),
     5RL(600),CRL(600),N,UBW,NJ,IX11,IX12,IY13,IX14,IX15,NROW,NUPW,
     6ST(360,360),D(600),
     1NLS,LN,IFLAG,IERR,ITRANS,ISTEP
      COMMON IDEP(100,9),NJD,IP(100),JROW(100,7),NRJT,IOR(600),JD,      ORD0004
     1INDEP,IDIAG(100),IRR,INR,INC,KTREC,IPAPT,IJK,KLM,ICROW,II,JJ,
     2S(6,600),LLD,STEMP(6,600),DUM(6,6),KUPW,NP,NFUL1,IP,T(600,6),
     3D1(600),IFUL1,JLIM,NF,XP,ZP,JTN(16),MJ(2,2),JF(24),ISJ(24,32)
      INTEGER UBW,RL,CRL                                                ORD0005
C
C   FORM IOR ARRAY, IOR(I)=NEW COLUMN/ROW POSITION IN REORDERED
C   K OF COLUMN/ROW I IN ORIGINAL K
C
      DO 420 I=1,N                                                      ORD0006
      IOR(I)=0                                                          ORD0007
  420 CONTINUE                                                          ORD0008
      IM=0                                                              ORD0009
      INDEP=N-JD                                                        ORD0010
      JM=INDEP                                                          ORD0011
      I=0                                                               ORD0012
  430 I=I+1                                                             ORD0013
      DO 440 J=1,NJD                                                    ORD0014
      LIM=2+IDEP(J,2)                                                   ORD0015
      DO 440 K=3,LIM                                                    ORD0016
      IF(IDEP(J,K).EQ.I) GO TO 450                                      ORD0017
  440 CONTINUE                                                          ORD0018
      IM=IM+1                                                           ORD0019
      IOR(I)=IM                                                         ORD0020
      IF(I.EQ.N) GO TO 460                                              ORD0021
      GO TO 430                                                         ORD0022
  450 JM=JM+1                                                           ORD0023
      IOR(I)=JM                                                         ORD0024
      IF(I.EQ.N) GO TO 460                                              ORD0025
      GO TO 430                                                         ORD0026
  460 RETURN                                                            ORD0027
      END                                                               ORD0028
```

```
*DECK UNPACK                                                            UNP0000
      SUBROUTINE UNPACK                                                 UNP0001
      COMMON DUM5(972),DUM6(39),IDUM3(44),IDUM4(14),IDUM5(315),DUM8(212)UNP0002
      COMMON IDUM1(602),DUM1(601),DUM2(1800),DUM3(365100),DUM4(4057),   UNP0003
     1IDUM2(905),
     5RL(600),CRL(600),N,UBW,NJ,IX11,IX12,IX13,IX14,IX15,NROW,NUBW,
     6ST(360,360),D(600),
     1NLS,LN,IFLAG,IESR,ITRANS,ISTEP
      COMMON IDEP(100,8),NJD,IR(100),JROW(100,7),NRJT,IOR(600),JD,      UNP0004
     1INDEP,IDIAG(100),IRR,INR,INC,KTREC,IPART,IJK,KLM,ICROW,II,JJ,
     2S(6,600),LLD,STEMP(6,600),DUM(6,6),KUBW,NP,NFUL1,IP,T(600,6),
     3D1(600),IFUL1,JLIM,NF,XP,ZP,JTN(16),MJ(2,2),JF(24),ISJ(24,32)
      INTEGER UBW,RL,CRL                                                UNP0005
      IREC=0                                                            UNP0006
      DO 1 II=1,NRJT                                                    UNP0007
C
C   ZERO OUT ARRAY USED TO STORE PARTITIONED ROW OF STIFFNESS MATRIX
C   .
C   ZERO OUT ARRAY USED TO STORE UNPACKED PARTITIONED ROW OF STIFFNESS MA
C                                                                 TRIX
C
      DO 38 I=1,6                                                       UNP0008
      DO 38 J=1,600                                                     UNP0009
      STEMP(I,J)=0.0                                                    UNP0010
   38 S(I,J)=0.0                                                        UNP0011
      IRII=IR(II)                                                       UNP0012
C
C   READ IN ENOUGH ROWS OF PACKED STIFFNESS MATRIX TO FORM ONE PARTITIONE
C                                                                 D ROW
C
      DO 2 I=1,IRII                                                     UNP0013
      IREC=IREC+1                                                       UNP0014
      READ(11"IREC)(S(I,J),J=1,UBW)                                     UNP0015
    2 CONTINUE                                                          UNP0016
      ICROW=ICROW+IR(II)                                                UNP0017
C
C   UNPACKING A PARTITIONED ROW OF S(I,J) INTO STEMP(I,J)
C
      J=0                                                               UNP0018
      LLD=0                                                             UNP0019
      K=1                                                               UNP0020
C
C   INR IS THE COLUMN NUMBER IN K OF A COLUMN IN PARTITIONED ROW II
C
      INR=INR+1                                                         UNP0021
C
C   IUBW IS THE LAST COLUMN WITH NONZERO VALUE IN ROW I OF STEMP(K,INC)
C   UBW=BAND WIDTH(FROM DIAGONAL TO LAST NONZERO ELEMENT IN K,INCLUSIVE)
```

```
C
      IUBW=UBW+INR-1
  480 J=J+1
      INC=INC+1
      STEMP(K,INC)=S(K,J)
      IF(INC.LT.IUBW) GO TO 480
      IF(K.EQ.IR(II)) GO TO 490
      K=K+1
      INR=INR+1
      IUBW=UBW+INR-1
      J=0
      INC=INR-1
      GO TO 480
  490 INC=ICROW
C
C  JJ=PARTITIONED COLUMN NUMBER NUMBER OF K
C  TRANSPOSE UNPACKED DIAGONAL,FILL IN SUBMATRICES TO LEFT OF DIAGONAL S
C                                                                UBMATRIX
C  IN UNPACKED PARTITIONED ROW, WRITE OUT SUBMATRICES TO RIGHT OF DIAGON
C                                                                AL
C  SUBMATRIX FOR USE LATER IN FILLING IN SUBMATRICES TO LEFT OF DIAGONAL
C  IN OTHER PARTITIONED ROWS
C
      DO 650 JJ=1,NRJT
      IF(II.LE.JJ) GO TO 570
      CALL LDIAG
      GO TO 650
  570 IF(II.EQ.JJ) GO TO 580
      CALL RDIAG
      GO TO 650
  580 CALL DIAG
      GO TO 650
  650 CONTINUE
      INR1=INR-IRII+1
      IJ=0
C
C  AFTER UNPACKING A PARTITIONED ROW,WRITE OUT EACH SEPARATE ROW TO DISK
C
      DO 3 I=INR1,INR
      IJ=IJ+1
      K=UBW+I-1
      IF(K.GT.N) K=N
      WRITE(11"I)(STEMP(IJ,J),J=1,K)
    3 CONTINUE
    1 CONTINUE
      RETURN
      END
```

```
*DECK LDIAG                                                           LDI0000
      SUBROUTINE LDIAG                                                LDI0001
      COMMON DUM5(972),DUM6(39),IDUM3(44),IDUM4(14),IDUM5(315),DUM8(212)LDI0002
      COMMON IDUM1(602),DUM1(601),DUM2(1800),DUM3(365100),DUM4(4057),   LDI0003
     1IDUM2(905),
     5RL(600),CRL(600),N,UBW,NJ,IX11,IX12,IX13,IX14,IX15,NROW,NUBW,
     6ST(360,360),D(600),
     1NLS,LN,IFLAG,IERR,ITRANS,ISTEP
      COMMON IDEP(100,8),NJD,IR(100),JROW(100,7),NRJT,IOR(600),JD,      LDI0004
     1INDEP,IDIAG(100),IRR,INR,INC,KTREC,IPART,IJK,KLM,ICROW,II,JJ,
     2S(6,600),LLD,STEMP(6,600),DUM(6,6),KUBW,NP,NFUL1,IP,T(600,6),
     3D1(600),IFUL1,JLIM,NF,XP,ZP,JTN(16),MJ(2,2),JF(24),ISJ(24,32)
      INTEGER UBW,RL,CRL                                              LDI0005
      IRII=IR(II)                                                     LDI0006
      IRJJ=IR(JJ)                                                     LDI0007
C
C  TO LEFT OF DIAGONAL
C
      LLD=0                                                           LDI0008
      IF(JJ.EQ.1) GO TO 510                                           LDI0009
      JJ1=JJ-1                                                        LDI0010
      DO 500 I=1,JJ1                                                  LDI0011
      LLD=LLD+IR(I)                                                   LDI0012
  500 CONTINUE                                                        LDI0013
  510 LL=LLD                                                          LDI0014
      II1=II-1                                                        LDI0015
      DO 520 I=JJ,II1                                                 LDI0016
      LL=LL+IR(I)                                                     LDI0017
  520 CONTINUE                                                        LDI0018
      IF((LL+1).GT.(UBW+LLD+IR(JJ)-1)) GO TO 565                      LDI0019
      DO 530 I=1,6                                                    LDI0020
      DO 530 J=1,6                                                    LDI0021
      DUM(I,J)=0.0                                                    LDI0022
  530 CONTINUE                                                        LDI0023
      IREC=IDIAG(JJ)+II-JJ                                            LDI0024
      READ(12'IREC)((DUM(K,L),K=1,IRJJ),L=1,IRII)                     LDI0025
      LL=0                                                            LDI0026
      IF(JJ.EQ.1) GO TO 550                                           LDI0027
      JJ1=JJ-1                                                        LDI0028
      DO 540 I=1,JJ1                                                  LDI0029
      LL=LL+IR(I)                                                     LDI0030
  540 CONTINUE                                                        LDI0031
  550 LSUM=LL                                                         LDI0032
      DO 560 I=1,IRII                                                 LDI0033
      LL=LSUM                                                         LDI0034
      DO 560 J=1,IRJJ                                                 LDI0035
      LL=LL+1                                                         LDI0036
      STEMP(I,LL)=DUM(J,I)                                            LDI0037
```

```
560 CONTINUE
565 RETURN
    END
```

```
*DECK RDIAG                                                             RDI0000
      SUBROUTINE RDIAG                                                  RDI0001
      COMMON DUM5(972),DUM6(39),IDUM3(44),IDUM4(14),IDUM5(315),DUM8(212)RDI0002
      COMMON IDUM1(602),DUM1(601),DUM2(1800),DUM3(365100),DUM4(4057),   RDI0003
     1IDUM2(905),
     5RL(600),CRL(600),N,UBW,NJ,IX11,IX12,IX13,IX14,IX15,NROW,NUBW,
     6ST(360,360),D(600),
     1NLS,LN,IFLAG,IERR,ITRANS,ISTED
      COMMON IDEP(100,8),NJD,IR(100),JROW(100,7),NRJT,IOR(600),JD,      RDI0004
     1INDEP,IDIAG(100),IRR,INR,INC,KTREC,IPART,IJK,KLM,ICROW,II,JJ,
     2S(6,600),LLD,STEMP(6,600),DUM(6,6),KUBW,NP,NFUL1,IP,T(600,6),
     3D1(600),IFUL1,JLIM,NF,XP,ZP,JTN(16),MJ(2,2),JF(24),ISJ(24,32)
      INTEGER UBW,RL,CRL                                                RDI0005
      IRII=IR(II)                                                       RDI0006
C
C  TO RIGHT OF DIAGONAL
C
      LL=0                                                              RDI0007
      JJ1=JJ-1                                                          RDI0008
      DO 600 I=1,JJ1                                                    RDI0009
      LL=LL+IR(I)                                                       RDI0010
  600 CONTINUE                                                          RDI0011
      IF((LL+1).GT.(UBW+LLD+IR(II)-1)) GO TO 610                        RDI0012
      LSUM=LL+IR(JJ)                                                    RDI0013
      LL=LL+1                                                           RDI0014
      IREC=IDIAG(II)+JJ-II                                              RDI0015
      WRITE(12"IREC)((STEMP(I,J),I=1,IRII),J=LL,LSUM)                   RDI0016
  610 RETURN                                                            RDI0017
      END                                                               RDI0018
```

```
*DECK DIAG                                                              DIA0000
      SUBROUTINE DIAG                                                   DIA0001
      COMMON DUM5(972),DUM6(39),IDUM3(44),IDUM4(14),IDUM5(315),DUM8(212)DIA0002
      COMMON IDUM1(602),DUM1(601),DUM2(1800),DUM3(365100),DUM4(4057),   DIA0003
     1IDUM2(905),
     5RL(600),CRL(600),N,UBW,NJ,IX11,IX12,IX13,IX14,IX15,NROW,NUBW,
     6ST(360,360),D(600),
     1NLS,LN,IFLAG,IERR,ITRANS,ISTED
      COMMON IDEF(100,6),NJD,IR(100),JROW(100,7),NRJT,IOR(600),JD,      DIA0004
     1INDEP,IDIAG(100),IRR,INR,INC,KTREC,IPART,IJK,KLM,ICROW,II,JJ,
     2S(6,600),LLD,STEMP(6,600),DUM(6,6),KUPW,NP,MFUL1,IP,T(600,6),
     3D1(600),IFUL1,JLIM,NF,XP,ZP,JTN(16),MJ(2,2),JF(24),ISJ(24,32)
      INTEGER UPW,RL,CRL                                                DIA0005
C
C  ON DIAGONAL
C
      LL=0                                                              DIA0006
      IF(JJ.EQ.1) GO TO 630                                             DIA0007
      JJ1=JJ-1                                                          DIA0008
      DO 620 I=1,JJ1                                                    DIA0009
      LL=LL+IR(I)                                                       DIA0010
  620 CONTINUE                                                          DIA0011
  630 LSUM=LL+IR(JJ)                                                    DIA0012
C
C  LLD=VALUE OF LL FOR DIAGONAL SUBMATRIX IN PARTITIONED ROW II
C
      LLD=LL                                                            DIA0013
      LL=LL+1                                                           DIA0014
      MM=0                                                              DIA0015
C
C  TRANSPOSE DIAGONAL SUBMATRIX
C
      LSUM1=LSUM-1                                                      DIA0016
      DO 640 J=LL,LSUM1                                                 DIA0017
      KK=J+1                                                            DIA0018
      MM=MM+1                                                           DIA0019
      IRII=IR(II)                                                       DIA0020
      DO 640 I=2,IRII                                                   DIA0021
      IF(I.LE.MM) GO TO 640                                             DIA0022
      STEMP(I,J)=STEMP(MM,KK)                                           DIA0023
      KK=KK+1                                                           DIA0024
  640 CONTINUE                                                          DIA0025
      RETURN                                                            DIA0026
      END                                                               DIA0027
```

```
*DECK ROWBND                                                            ROW0000
      SUBROUTINE ROWBND                                                 ROW0001
      COMMON DUM5(972),DUM6(39),IDUM3(44),IDUM4(14),IDUM5(315),DUM8(212)ROW0002
      COMMON IDUM1(602),DUM1(601),DUM2(1800),DUM3(365100),DUM4(4057),   ROW0003
     1IDUM2(905),
     5RL(600),CRL(600),N,UBW,NJ,IX11,IX12,IX13,IX14,IX15,NROW,NUBW,
     6ST(360,360),D(600),
     1NLS,LN,IFLAG,IEPR,ITRANS,ISTED
      COMMON IDEP(100,8),NJD,IR(100),JROW(100,7),NRJT,IOR(600),JD,      ROW0004
     1INDEP,IDIAG(100),IRR,INR,INC,KTREC,IPART,IJK,KLM,ICROW,II,JJ,
     2S(6,600),LLD,STEMP(6,600),DUM(6,6),KUBW,NP,NFUL1,IP,T(600,6),
     3D1(600),IFUL1,JLIM,NF,XP,ZP,JTN(16),MJ(2,2),JF(24),ISJ(24,32)
      INTEGER UBW,RL,CRL                                                ROW0005
      LL=0                                                              ROW0006
C
C   DETERMINE BAND WIDTH(INCLUDING DIAGONAL)OF PARTITIONED REORDERED ROW
C                                                                II OF K
C   THIS VALUE IS KUBW
C
      IF(II.LE.1) GO TO 720                                             ROW0007
      II1=II-1                                                          ROW0008
      DO 710 I=1,II1                                                    ROW0009
      LL=LL+IR(I)                                                       ROW0010
  710 CONTINUE                                                          ROW0011
  720 MAXCOL=1                                                          ROW0012
      KUBW=0                                                            ROW0013
      IRII=IR(II)                                                       ROW0014
      DO 770 I=1,IRII                                                   ROW0015
C
C   LL=DIAGONAL ELEMENT COLUMN NUMBER IN REORDERED K
C
      LL=LL+1                                                           ROW0016
C
C   N=TOTAL NUMBER OF UNKNOWN DISPLACEMENT COMPONENTS
C   (DEPENDENT AND INDEPENDENT ORIGINAL JOINT DISPLACEMENT COMPONENTS)
C
      DO 770 J=LL,N                                                     ROW0017
      DO 725 K=1,N                                                      ROW0018
      IF(IOR(K).EQ.J) GO TO 728                                         ROW0019
  725 CONTINUE                                                          ROW0020
  728 IF(J.EQ.LL) GO TO 740                                             ROW0021
      IF(ABS(STEMP(I,K)).GT..0001) GO TO 750                            ROW0022
  730 IF(MAXCOL.GT.KUBW) GO TO 760                                      ROW0023
      GO TO 770                                                         ROW0024
C
C   ON DIAGONAL
C
  740 IF(STEMP(I,K).LE..0001) GO TO 780                                 ROW0025
```

```
750 MAXCOL=J-LL+1
    GO TO 730
760 KUBW=MAXCOL
    GO TO 770
770 CONTINUE
    GO TO 790
780 IERR=1
    WRITE(6,785)
785 FORMAT(1X,"REORDERED MATRIX HAS A ZERO ON THE DIAGONAL"/)
790 RETURN
    END
```

```
*DECK RPKKT
      SUBROUTINE RPKKT
      COMMON DUM5(972),DUM6(39),IDUM3(44),IDUM4(14),IDUM5(315),DUM8(212)
      COMMON IDUM1(602),DUM1(601),DUM2(1800),DUM3(365100),DUM4(4057),
     1IDUM2(905),
     5RL(600),CRL(600),N,UBW,NJ,IX11,IX12,IX13,IX14,IX15,NROW,NUBW,
     6ST(360,360),D(600),
     1NLS,LN,IFLAG,IERR,ITRANS,ISTED
      COMMON IDFR(100,8),NJD,IR(100),JROW(100,7),NRJT,IOR(600),JD,
     1INDEP,IDIAG(100),IRR,INR,INC,KTREC,IPART,IJK,KLM,ICROW,II,JJ,
     2S(6,600),LLD,STEMP(6,600),DUM(6,6),KURL,NP,NFUL1,IP,T(600,6),
     3D1(600),IFUL1,JLIF,NF,XP,ZP,JTN(16),MJ(2,2),JF(24),ISJ(24,32)
      INTEGER UBW,RL,CRL
C
C  ZERO OUT ARRAY USED TO TEMPORARILY STORE PARTITIONED ROW OF REDUCED
C  STIFFNESS MATRIX
C
      DO 780 I=1,6
      DO 780 J=1,600
      S(I,J)=0.0
  780 CONTINUE
      IF(IPART.EQ.1) GO TO 800
C
C  PACK K11 OF REORDERED K INTO S(I,J) ASSOCIATED WITH PARTITIONED ROW I
C                                                                      I
C
      K=INR-IR(II)
      IRJK=IR(II)-IJK
      DO 790 I=1,IRJK
      K=K+1
      DO 790 J=K,INDEP
      IV=J-K+1
      DO 785 KM=1,N
      IF(IOR(KM).EQ.J) GO TO 787
  785 CONTINUE
  787 S(I,IV)=STEMP(I,KM)
  790 CONTINUE
C
C  PARTITIONED ROW II BELOW HORIZONTAL PARTITION IN REORDERED K
C
C
C  NP=NUMBER OF COLUMN PARTITIONS IN T(PASSED FROM T ROUTINE)
C
  800 DO 890 I1=1,NP
      NA=6
      IA=6
      IF(I1.EQ.NP) GO TO 820
  810 LL=0
```

```
C
C     READ IN ONE COMPLETE PARTITIONED COLUMN OF T,COLUMN I1
C     IP=NUMBER OF ROW PARTITIONS IN T(PASSED FROM T ROUTINE)
C
         GO TO 830
C
C     NFUL1=NUMBER OF ABSOLUTE COLUMNS IN LAST COLUMN PARTITION OF T
C
  820 NA=NFUL1
      GO TO 810
  830 DO 860 J1=1,IP
      IF(J1.EQ.IP) GO TO 850
  840 KK=LL+1
      LL=LL+IA
      ITREC=((J1-1)*NP)+I1
      READ(13'ITREC)((T(IK,IJ),IK=KK,LL),IJ=1,NA)
      GO TO 860
C
C     IFUL1=NUMBER OF ROWS IN LAST PARTITIONED ROW OF T
C
  850 IA=IFUL1
      GO TO 840
  860 CONTINUE
C
C     GENERATE K12T AND/OR K22T ASSOCIATED WITH PARTITIONED ROW II IN PACKE
C                                                                        D
C     FORM
C
      IRII=IP(II)
      DO 885 I=1,IRII
      IBELOW=0
C
C     KK=NUMBER OF ELEMENTS IN ROW I OF PARTITIONED ROW II IN REORDERED K12
C                                                                   OR K22
C     UP TO THE BAND OF REORDERED K+INDEP
C
      KK=KURW+INR-IR(II)+I-1
      IF(KK.GT.N) KK=N
      DO 880 ICOLT=1,NA
      IF(KK.LE.0) GO TO 880
      SUM=0.0
      INDEP1=INDEP+1
      DO 870 L=INDEP1,KK
      IROWT=L-INDEP
      DO 865 J=1,N
      IF(IOP(J).EQ.L) GO TO 868
  865 CONTINUE
  868 SUM=SUM+STEMP(I,J)*T(IROWT,ICOLT)
```

```
  870 CONTINUE                                                          RPK0055
      K=INR-IR(II)+I                                                    RPK0056
      JPAK=((I1-1)*6)+ICOLT+INDEP-K+1                                   RPK0057
C
C   ABOVE PARTITION
C
      IF(I.LE.(IR(II)-INR+INDEP)) GO TO 878                             RPK0058
C
C   ROW IS BELOW K11 K12 PARTITION
C
      IBELOW=IBELOW+1                                                   RPK0059
      JPAK=INDEP+IBELOW+((I1-1)*6)                                      RPK0060
C
CC   S(I,J)=THE PACKED FORM OF CONDENSED REORDERED K FOR PARTITIONED ROW
C                                                                 II
C
  878 S(I,JPAK)=SUM                                                     RPK0061
  880 CONTINUE                                                          RPK0062
  885 CONTINUE                                                          RPK0063
  890 CONTINUE                                                          RPK0064
      RETURN                                                            RPK0065
      END                                                               RPK0066
```

```
*DECK STORE
      SUBROUTINE STORE
      COMMON DUM5(972),DUM6(39),IDUM3(44),IDUM4(14),IDUM5(315),DUM8(212)
      COMMON IDUM1(602),DUM1(601),DUM2(1800),DUM3(365100),DUM4(4057),
     1IDUM2(905),
     5RL(600),CRL(600),N,UBW,NJ,IX11,IX12,IY13,IX14,IX15,NROW,NURW,
     6ST(360,360),D(600),
     1NLS,LN,IFLAG,IERR,ITRANS,ISTED
      COMMON IDEP(100,8),NJC,IR(100),JROW(100,7),NRJT,ICR(600),JD,
     1INDEP,IDIAG(100),IRR,INR,INC,KTREC,IPART,IJK,KLM,ICROW,II,JJ,
     2S(6,600),LLD,STFMP(6,600),DUM(6,6),KURW,NP,NFUL1,IP,T(600,6),
     3D1(600),IFUL1,JLIM,NF,XP,ZP,JTN(16),MJ(2,2),JF(24),ISJ(24,32)
      INTEGER UBW,RL,CRL
      IRII=IR(II)
      IF(IPART.EQ.1) GO TO 940
C
C  PARTITIONED ROW II OF REDUCED STIFFNESS MATRIX IS STORED IN ITS FINAL
C  POSITION IN ARRAY ST(I,J) IF IT IS ABOVE THE HORIZONTAL PARTITION
C
      I=0
      INR1=INR-IR(II)+1
      INJK=INR-IJK
      DO 900 K=INR1,INJK
      JLIM1=JLIM-K+1
      I=I+1
      DO 900 J=1,JLIM1
      ST(K,J)=S(I,J)
  900 CONTINUE
      IF(INR.LE.INDEP) GO TO 980
C
C  RECORDS OF K22T ASSOCIATED WITH PARTITIONED ROW II ARE WRITTEN TO STO
C                                                                RAGE
C
C
C
C  PARTITIONED ROW II STRADDLES HORIZONTAL PARTITION
C
      NA=6
      LL=INDEP
      DO 930 I=1,NP
      IF(I.EQ.NP) GO TO 920
  910 KK=LL+1
      LL=LL+NA
      KTREC=I
      IRIJK=IR(II)-IJK+1
      WRITE(14'KTREC)((S(K,J),K=IRIJK,IRII),J=KK,LL)
      GO TO 930
  920 NA=NFUL1
      GO TO 910
```

```
  930 CONTINUE
      GO TO 980
C
C  THIS PARTITIONED ROW IS COMPLETELY BELOW HORIZONTAL PARTITION
C
  940 NA=6
      LL=INDEP
      DO 970 I=1,NP
      IF(I.EQ.NP) GO TO 960
  950 KTREC=KTREC+1
      KK=LL+1
      LL=LL+NA
      WRITE(14'KTREC)((S(K,J),K=1,IRII),J=KK,LL)
      GO TO 970
  960 NA=NFUL1
      GO TO 950
  970 CONTINUE
      GO TO 980
C
C  IF RECORDS OF K11 AND K12T ARE NEEDED THEY CAN BE FORMED HERE
C
  980 RETURN
      END
```

```
*DECK TTRKT                                                             TTR0000
      SUBROUTINE TTRKT                                                  TTR0001
      COMMON DUM5(972),DUM6(39),IDUM3(44),IDUM4(14),IDUM5(315),DUM8(212)TTR0002
      COMMON IDUM1(602),DUM1(601),DUM2(1800),DUM3(365100),DUM4(4057),   TTR0003
     1IDUM2(905),
     5RL(600),CRL(600),M,UBW,NJ,IX11,IX12,IX13,IX14,IX15,NROW,NUBW,
     6ST(360,360),D(600),
     1NLS,LN,IFLAG,IERR,ITRANS,ISTED
      COMMON IDEP(100,8),NJD,IR(100),JROW(100,7),NRJT,IOR(600),JD,      TTR0004
     1INDEP,IDIAG(100),IRN,INR,INC,KTREC,IPART,IJK,KLM,ICROW,II,JJ,
     2S(6,600),LLD,STEMP(6,600),DUM(6,6),KUBW,NP,NFUL1,IP,T(600,6),
     3D1(600),IFUL1,JLIN,NF,XP,ZP,JTN(16),MJ(2,2),JF(24),ISJ(24,32)
      INTEGER UBW,RL,CRL                                                TTR0005
C
C  FORM TTRKT AND STORE RESULTS IN FINAL REDUCED MATRIX ARRAY ST(I,J)
C
      KKK=INDEP                                                         TTR0006
C
C  KP=NUMBER OF ROW PARTITIONS OF K22T
C
      IF(KLM.EQ.0) GO TO 1000                                           TTR0007
      IF(IJK.EQ.0) GO TO 1005                                           TTR0008
      KP=NRJT-KLM+2                                                     TTR0009
      GO TO 1010                                                        TTR0010
 1000 KP=1                                                              TTR0011
      GO TO 1010                                                        TTR0012
 1005 KP=NRJT-KLM+1                                                     TTR0013
      GO TO 1010                                                        TTR0014
C
C  INCREMENT ON PARTITIONED COLUMNS OF T OR PARTITIONED ROWS OF T TRANSP
C                                                              OSE
C
 1010 DO 1200 I1=1,NP                                                   TTR0015
      IA=6                                                              TTR0016
      NA=6                                                              TTR0017
      IF(I1.EQ.NP) GO TO 1030                                           TTR0018
 1020 LL=0                                                              TTR0019
      GO TO 1040                                                        TTR0020
 1030 NA=NFUL1                                                          TTR0021
      GO TO 1020                                                        TTR0022
 1040 DO 1070 J1=1,IP                                                   TTR0023
      IF(J1.EQ.IP) GO TO 1060                                           TTR0024
 1050 KK=LL+1                                                           TTR0025
      LL=LL+IA                                                          TTR0026
      ITREC=((J1-1)*NP)+I1                                              TTR0027
      READ(13"ITREC)((T(IK,IJ),IK=KK,LL),IJ=1,NA)                       TTR0028
      GO TO 1070                                                        TTR0029
 1060 IA=IFUL1                                                          TTR0030
```

```
      GO TO 1050                                                        TTR0031
 1070 CONTINUE                                                          TTR0032
      LLL=0                                                             TTR0033
C
C  KA=NUMBER OF COLUMNS IN THE I2 PARTITIONED COLUMN OF K22T
C
      KA=6                                                              TTR0034
C
C  READ IN THE I2 PARTITIONED COLUMN OF K22T
C
      DO 1190 I2=1,NP                                                   TTR0035
      IF(I2.EQ.NP) GO TO 1090                                           TTR0036
 1080 LL=0                                                              TTR0037
C
C  INCREMENT ON PARTITIONED ROW OF K22T IN PARTITIONED COLUMN OF K22T
C
      GO TO 1100                                                        TTR0038
 1090 KA=NFUL1                                                          TTR0039
      GO TO 1080                                                        TTR0040
 1100 DO 1150 J2=1,KP                                                   TTR0041
      IF(IJK.NE.0) GO TO 1130                                           TTR0042
      K2=J2+KLM-1                                                       TTR0043
 1110 IA=IR(K2)                                                         TTR0044
 1120 KK=LL+1                                                           TTR0045
      LL=LL+IA                                                          TTR0046
      KTREC=((J2-1)*NP)+I2                                              TTR0047
C
C  READ IN COLUMNS OF K22T,STORE IN S(I,J)
C
      READ(14'KTREC)((S(IJ,IK),IK=KK,LL),IJ=1,KA)                       TTR0048
      GO TO 1150                                                        TTR0049
 1130 IF(J2.GT.1) GO TO 1140                                            TTR0050
      IA=IJK                                                            TTR0051
      GO TO 1120                                                        TTR0052
 1140 K2=J2+KLM-2                                                       TTR0053
      GO TO 1110                                                        TTR0054
 1150 CONTINUE                                                          TTR0055
C
C  INCREMENT ON ROWS OF PARTITIONED ROW OF T TRANSPOSE
C  NA=NUMBER OF COLUMNS IN A PARTITIONED COLUMN OF T
C  OR NUMBER OF ROWS IN PARTITIONED ROW OF T TRANSPOSE
C
      DO 1180 K=1,NA                                                    TTR0056
C
C  KKK=ABSOLUTE ROW NUMBER IN PACKED REDUCED K,ST(I,J)
C
      KKK=KKK+1                                                         TTR0057
C
```

```
C   KA=NUMBER OF COLUMNS IN A PARTITIONED COLUMN OF K22T, INCREMENT ON
C   COLUMNS OF K22T
C
       DO 1170 J=1,KA                                                   TTR0058
       SUM=0.0                                                          TTR0059
       LLL=LLL+1                                                        TTR0060
C
C   SKIP COMPUTATION OF ELEMENTS BELOW DIAGONAL OF
C   T TRANSPOSE TIMES K22 TIMES T
C
       IF((KKK-INDEP).GT.LLL) GO TO 1170                                TTR0061
C
C   JD=NUMBER OF DEPENDENT DISPLACEMENT COMPONENTS
C
       DO 1160 I=1,JD                                                   TTR0062
       SUM=SUM+T(I,K)*S(J,I)                                            TTR0063
 1160  CONTINUE                                                         TTR0064
       L=INDEP+LLL-KKK+1                                                TTR0065
       ST(KKK,L)=SUM                                                    TTR0066
 1170  CONTINUE                                                         TTR0067
       IF(K.EQ.NA) GO TO 1180                                           TTR0068
       LLL=LLL-KA                                                       TTR0069
       GO TO 1180                                                       TTR0070
 1180  CONTINUE                                                         TTR0071
       KKK=KKK-NA                                                       TTR0072
 1190  CONTINUE                                                         TTR0073
       KKK=KKK+NA                                                       TTR0074
 1200  CONTINUE                                                         TTR0075
       NROW=INDEP+IRR                                                   TTR0076
       NUBW=0                                                           TTR0077
       DO 1340 I=1,NROW                                                 TTR0078
       DO 1340 J=1,NROW                                                 TTR0079
       IF(ABS(ST(I,J)).GT..0001) GO TO 1330                             TTR0080
       GO TO 1340                                                       TTR0081
 1330  IF(J.GT.NUBW) NUBW=J                                             TTR0082
 1340  CONTINUE                                                         TTR0083
       RETURN                                                           TTR0084
       END                                                              TTR0085
```

```
*DECK RLOADV                                                           RLO00000
      SUBROUTINE RLOADV                                                RLO00001
      COMMON DUM5(972),DUM6(39),IDUM3(44),IDUM4(14),IDUM5(315),DUM8(212)RLO00002
      COMMON IDUM1(602),DUM1(601),DUM2(1800),DUM3(365100),DUM4(4057),  RLO00003
     1IDUM2(905),
     5RL(600),CRL(600),N,UBW,NJ,IX11,IX12,IX13,IX14,IX15,NROW,NUBW,
     6ST(360,360),D(600),
     1NLS,LN,IFLAG,IERR,ITRANS,ISTED
      COMMON IDEP(100,8),NJD,IR(100),JROW(100,7),NRJT,IOR(600),JD,     RLO00004
     1INDEP,IDIAG(100),IRR,INR,INC,KTREC,IPART,IJK,KLM,ICROW,II,JJ,
     2S(6,600),LLD,STEMP(6,600),DUM(6,6),KURV,NP,NFUL1,IP,T(600,6),
     3D1(600),IFUL1,JLIM,NF,XP,ZP,JTN(16),MJ(2,2),JF(24),ISJ(24,32)
      INTEGER UBW,RL,CRL                                               RLO00005
C
C  FORM REDUCED LOAD VECTOR
C
C
C  READ IN LOAD VECTOR LN
C
      READ(15"LN) (S(1,K),K=1,N)                                       RLO00006
      L=INDEP                                                          RLO00007
C
C  READ IN PARTITIONED COLUMNS OF T OR PARTITIONED ROWS OF T TRANSPOSE
C
      DO 1310 I1=1,NP                                                  RLO00008
      IA=6                                                             RLO00009
      NA=6                                                             RLO00010
      IF(I1.EQ.NP) GO TO 1220                                          RLO00011
 1210 LL=0                                                             RLO00012
      GO TO 1230                                                       RLO00013
 1220 NA=NFUL1                                                         RLO00014
      GO TO 1210                                                       RLO00015
 1230 DO 1260 J1=1,IP                                                  RLO00016
      IF(J1.EQ.IP) GO TO 1250                                          RLO00017
 1240 KK=LL+1                                                          RLO00018
      LL=LL+IA                                                         RLO00019
      ITREC=((J1-1)*NP)+I1                                             RLO00020
      READ(13"ITREC)((T(IK,IJ),IK=KK,LL),IJ=1,NA)                      RLO00021
      GO TO 1260                                                       RLO00022
 1250 IA=IFUL1                                                         RLO00023
      GO TO 1240                                                       RLO00024
 1260 CONTINUE                                                         RLO00025
      IF(L.GT.INDEP) GO TO 1290                                        RLO00026
C
C  REORDER LOAD VECTOR
C
      DO 1270 I=1,N                                                    RLO00027
      DO 1265 KM=1,N                                                   RLO00028
```

```
      IF(IOR(KM).EQ.I) GO TO 1268                                       RLO00029
 1265 CONTINUE                                                          RLO00030
 1268 S(2,I)=S(1,KM)                                                    RLO00031
 1270 CONTINUE                                                          RLO00032
C
C  TRANSFER FIRST INDEP ELEMENTS OF LOAD VECTOR P1
C
      IF(INDEP.EQ.0) GO TO 1290                                         RLO00033
      DO 1280 I=1,INDEP                                                 RLO00034
      S(3,I)=S(2,I)                                                     RLO00035
 1280 CONTINUE                                                          RLO00036
C
C  NA=NUMBER OF ROWS IN PARTITIONED ROW OF T TRANSPOSE
C
 1290 DO 1300 IL=1,NA                                                   RLO00037
      K=INDEP                                                           RLO00038
      L=L+1                                                             RLO00039
      S(3,L)=0.0                                                        RLO00040
      DO 1300 IK=1,JD                                                   RLO00041
      K=K+1                                                             RLO00042
C
C  MULTIPLY T TRANSPOSE BY P2
C
      S(3,L)=T(IK,IL)*S(2,K)+S(3,L)                                     RLO00043
 1300 CONTINUE                                                          RLO00044
 1310 CONTINUE                                                          RLO00045
C
C  WRITE P1 AND T TRANSPOSE TIMES P2 OUT TO STORAGE
C
      LN1=LN+NLS                                                        RLO00046
      WRITE(15"LN1) (S(3,K),K=1,L)                                      RLO00047
      RETURN                                                            RLO00048
      END                                                               RLO00049
```

```
*DECK KFN                                                                    KFN0000
      FUNCTION KFN(IJ,IK)                                                    KFN0001
      COMMON DUM5(972),DUM6(39),IDUM3(44),IDUMA(14),IDUM5(315),DUM8(212)KFN0002
      COMMON IDUM1(602),DUM1(601),DUM2(1800),DUM3(365100),DUM4(4057),   KFN0003
     1IDUM2(905),
     5RL(600),CRL(600),N,UBW,NJ,IX11,IX12,IX13,IX14,IX15,NROW,NUBW,
     6ST(360,360),D(600),
     1NLS,LN,IFLAG,IERR,ITRANS,ISTED
      COMMON IDEP(100,8),NJD,IR(100),JROW(100,7),NRJT,IOR(600),JD,         KFN0004
     1INDEP,IDIAG(100),IRR,INR,INC,KTREC,IPART,IJK,KLM,ICROW,II,JJ,
     2S(6,600),LLD,STEMP(6,600),DUM(6,6),KUBW,NP,NFUL1,IP,T(600,6),
     3D1(600),IFUL1,JLIM,NF,XP,ZP,JTN(16),MJ(2,2),JF(24),ISJ(24,32)
      INTEGER UBW,RL,CRL                                                     KFN0005
      KFN=(6*(IDEP(IJ,1)-1))+IDEP(IJ,IK)                                    KFN0006
      RETURN                                                                 KFN0007
      END                                                                    KFN0008
```

```
*DECK EXPRD                                                              EXP0000
      SUBROUTINE EXPRD                                                   EXP0001
      COMMON DUM5(972),DUM6(39),IDUM3(44),IDUM4(14),IDUM5(315),DUM8(212)EXP0002
      COMMON IDUM1(602),DUM1(601),DUM2(1800),DUM3(365100),DUM4(4057),    EXP0003
     1IDUM2(905),
     5RL(600),CRL(600),N,UBW,NJ,IX11,IX12,IX13,IX14,IY15,MROW,NUBW,
     6ST(360,360),D(600),
     1NLS,LN,IFLAG,IERR,ITRANS,ISTEP
      COMMON IDEP(100,8),NJD,IP(100),JRCV(100,7),NPJT,IOR(600),JD,       EXP0004
     1INDEP,IDIAC(100),IRR,INR,INC,KTREC,IPART,IJK,KLM,ICROW,II,JJ,
     2S(6,600),LLD,STEMP(6,600),DUM(6,6),KUBW,NP,NFUL1,IP,T(600,6),
     3D1(600),IFUL1,JLIM,NF,XP,ZP,JTN(16),MJ(2,2),JF(24),ISJ(24,32)
      INTEGER UBW,PL,CRL                                                 EXP0005
C
C  EXPAND AND REORDER REDUCED DISPLACEMENT VECTOR
C
      ID=0                                                               EXP0006
      IF(INDEP.EQ.0) GO TO 1355                                          EXP0007
      DO 1350 I=1,INDEP                                                  EXP0008
      ID=ID+1                                                            EXP0009
      D1(I)=D(I)                                                         EXP0010
 1350 CONTINUE                                                           EXP0011
C
C  READ IN ONE PARTITIONED ROW OF T
C
 1355 DO 1440 I1=1,IP                                                    EXP0012
      NA=6                                                               EXP0013
      IA=6                                                               EXP0014
      IF(I1.EQ.IP) GO TO 1370                                            EXP0015
 1360 LL=0                                                               EXP0016
      GO TO 1380                                                         EXP0017
 1370 IA=IFUL1                                                           EXP0018
      GO TO 1360                                                         EXP0019
 1380 DO 1410 J1=1,NP                                                    EXP0020
      IF(J1.EQ.NP) GO TO 1400                                            EXP0021
 1390 KK=LL+1                                                            EXP0022
      LL=LL+NA                                                           EXP0023
      ITREC=((I1-1)*NP)+J1                                               EXP0024
      READ(13"ITREC)((S(IK,IJ),IK=1,IA),IJ=KK,LL)                        EXP0025
      GO TO 1410                                                         EXP0026
 1400 NA=NFUL1                                                           EXP0027
      GO TO 1390                                                         EXP0028
 1410 CONTINUE                                                           EXP0029
C
C  MULTIPLY T TIMES THE NUMBER OF DISPLACEMENT MEASURES,JD
C
      DO 1430 I=1,IA                                                     EXP0030
      SUM=0.0                                                            EXP0031
```

```
      K=INDEP                                                           EXP0032
      ID=ID+1                                                           EXP0033
      DO 1420 J=1,IRR                                                   EXP0034
      K=K+1                                                             EXP0035
      SUM=SUM+S(I,J)*D(K)                                               EXP0036
 1420 CONTINUE                                                          EXP0037
      D1(ID)=SUM                                                        EXP0038
 1430 CONTINUE                                                          EXP0039
 1440 CONTINUE                                                          EXP0040
C
C  REARRANGE DISPLACEMENTS BACK INTO THEIR ORIGINAL ORDER
C
      DO 1460 I=1,N                                                     EXP0041
      DO 1450 J=1,N                                                     EXP0042
      IF(IOR(J).NE.I) GO TO 1450                                        EXP0043
      D(J)=D1(I)                                                        EXP0044
      GO TO 1460                                                        EXP0045
 1450 CONTINUE                                                          EXP0046
 1460 CONTINUE                                                          EXP0047
      RETURN                                                            EXP0048
      END                                                               EXP0049
```

```
*DECK FORMT                                                              FOR0000
      SUBROUTINE FORMT                                                   FOR0001
      COMMON DUM5(972),DUM6(39),IDUM3(44),IDUM4(14),IDUM5(315),DUM8(212)FOR0002
      COMMON IDUM1(602),DUM1(601),DUM2(1800),DUM3(365100),DUM4(4057),   FOR0003
     1IDUM2(905),
     5RL(600),CPL(600),N,UBW,NJ,IX11,IX12,IY13,IY14,IY15,NROW,NUBW,
     6ST(360,360),D(600),
     1NLS,LN,IFLAG,IERR,ITRANS,ISTEP
      COMMON IDEP(100,8),NJD,IR(100),JROW(100,7),NRJT,IOR(600),JD,       FOR0004
     1INDEP,IDIAG(100),IRF,INR,INC,KTREC,IPART,IJK,KLM,ICROW,II,JJ,
     2S(6,600),LLD,STEMP(6,600),DUM(6,6),KURW,NP,NFUL1,IP,T(600,6),
     3D1(600),IFUL1,JLIM,NF,XP,ZP,JTN(16),MJ(2,2),JF(24),ISJ(24,32)
      INTEGER UBW,RL,CRL                                                 FOR0005
      IF(ITRANS.EQ.0) GO TO 45                                           FOR0006
C
C  INTERNALLY DETERMINE THE CORRECT PARTITIONING OF T
C
      NFUL1=6                                                            FOR0007
      IFUL1=6                                                            FOR0008
      NP=NF                                                              FOR0009
      IP=NJD                                                             FOR0010
      IF(ITRANS.EQ.1) GO TO 245                                          FOR0011
      NP=(NF*3)/6                                                        FOR0012
      XNP=(NF*3.)/6.                                                     FOR0013
      IF((XNP-NP).LE..0001) GO TO 235                                    FOR0014
      NP=NP+1                                                            FOR0015
      NFUL1=3                                                            FOR0016
  235 IP=(NJD*3)/6                                                       FOR0017
      XIP=(NJD*3.)/6.                                                    FOR0018
      IF((XIP-IP).LE..0001) GO TO 245                                    FOR0019
      IP=IP+1                                                            FOR0020
      IFUL1=3                                                            FOR0021
  245 IRR=NF*6                                                           FOR0022
      IF(ITRANS.EQ.1) GO TO 255                                          FOR0023
      IRR=NF*3                                                           FOR0024
  255 IODD=0                                                             FOR0025
      IA=6                                                               FOR0026
      JREC=0                                                             FOR0027
      NJP=0                                                              FOR0028
      JTMIN=1                                                            FOR0029
  265 DO 325 K=1,ITRANS                                                  FOR0030
      NJP=NJP+1                                                          FOR0031
  270 DO 285 I=1,NF                                                      FOR0032
      JFI=JF(I)                                                          FOR0033
      DO 275 J=1,JFI                                                     FOR0034
      IF(ISJ(I,J).EQ.JTMIN) GO TO 295                                    FOR0035
  275 CONTINUE                                                           FOR0036
  285 CONTINUE                                                           FOR0037
```

```
      JTMIN=JTMIN+1
      GO TO 270
  295 MJ(K,1)=JTMIN
      MJ(K,2)=I
      IF(NJP.EQ.NJD) GO TO 305
      JTMIN=JTMIN+1
      GO TO 325
  305 IF(K.EQ.1.AND.ITRANS.EQ.2) GO TO 315
      GO TO 335
  315 IODD=1
      IA=IFUL1
      GO TO 335
  325 CONTINUE
C
C  ZERO OUT ARRAY USED FOR TEMPORARY STORAGE OF T MATRIX
C
  335 DO 345 I=1,IA
      DO 345 J=1,600
      S(I,J)=0.0
  345 CONTINUE
      DO 355 K=1,ITRANS
      MJK1=MJ(K,1)
      READ(20"MJK1) XI,YI,ZI
      IF(ITRANS.EQ.1) GO TO 365
      ICOL=MJ(K,2)*3
      ITA=3*(K-1)
      S(1+ITA,ICOL)=ZI-ZP
      S(2+ITA,ICOL)=-(XI-XP)
      S(3+ITA,ICOL)=1.0
      S(2+ITA,ICOL-1)=1.0
      S(1+ITA,ICOL-2)=1.0
      IF(NJP.EQ.NJD.AND.IODD.EQ.1) GO TO 375
  355 CONTINUE
      GO TO 375
  365 ICOL=MJ(K,2)*6
      S(2,ICOL)=XI-XP
      S(6,ICOL)=1.0
      S(1,ICOL-1)=ZI-ZP
      S(3,ICOL-1)=-(XI-XP)
      S(5,ICOL-1)=1.0
      S(2,ICOL-2)=-(ZI-ZP)
      S(4,ICOL-2)=1.0
      S(3,ICOL-3)=1.0
      S(2,ICOL-4)=1.0
      S(1,ICOL-5)=1.0
      GO TO 375
  375 NA=6
      LL=0
```

```
      DO 405 J1=1,NP
      JREC=JREC+1
      IF(J1.EQ.NP) GO TO 385
  380 KK=LL+1
      LL=LL+NA
      GO TO 395
  385 NA=NFUL1
      GO TO 380
  395 WRITE(13'JREC) ((S(I,J),I=1,IA),J=KK,LL)
  405 CONTINUE
      IF(NJP.LT.NJD) GO TO 265
      GO TO 415
C
C  INPUT T MATRIX BY RECORD
C
   45 READ(5,50) IRR,IFUL1,NFUL1,IP,NP
   50 FORMAT(5I5)
      JREC=0
      IF(IP.EQ.1) GO TO 5
      IP1=IP-1
      DO 1 KROW=1,IP1
      IF(NP.EQ.1) GO TO 6
      NP1=NP-1
      DO 2 KCOL=1,NP1
      READ(5,100)((T(I,J),I=1,6),J=1,6)
      JREC=JREC+1
      WRITE(13'JREC)((T(I,J),I=1,6),J=1,6)
    2 CONTINUE
    6 READ(5,100)((T(I,J),I=1,6),J=1,NFUL1)
      JREC=JREC+1
      WRITE(13'JREC)((T(I,J),I=1,6),J=1,NFUL1)
    1 CONTINUE
    5 IF(NP.EQ.1) GO TO 7
      DO 3 KCOL=1,NP1
      READ(5,100)((T(I,J),I=1,IFUL1),J=1,6)
      JREC=JREC+1
      WRITE(13'JREC)((T(I,J),I=1,IFUL1),J=1,6)
    3 CONTINUE
    7 READ(5,100)((T(I,J),I=1,IFUL1),J=1,NFUL1)
      JREC=JREC+1
      WRITE(13'JREC)((T(I,J),I=1,IFUL1),J=1,NFUL1)
  100 FORMAT(6F10.3)
  415 RETURN
      END
```

```
*DECK MODFR2                                                               MOD0000
      SUBROUTINE MODFR2                                                    MOD0001
      COMMON DUM5(572),DUM6(39),IDUM3(44),IDUM4(11),                       MOD0002
     1IX16,IX17,IX18,IX19,IX20,IX21,IX22,IX23,IX24,IX25,IX26,IX27,
     2IDUM7(306),
     5COLID1(48),COLID2(48),BMID1(38),BMID2(38),BRID1(20),BRID2(20)
      COMMON ITAB(300),ISEC(300),IFIRST,M,E,DJ(600),                       MOD0003
     1X(100),Y(100),Z(100),L(300),AX(300),IX(300),IY(300),IZ(300),
     2AR(600),A(600),AC(600),AE(600),S(600,600),R(360,9),
     3AML(300,12),SM(12,12),SMD(12,12),SMR(12,12),AM(12),AMD(12),G,
     4JJ(300),JK(300),LML(300),JE,KE,TS,NDJ,NR,
     5RL(600),CRL(600),N,UBW,NJ,IX11,IX12,IX13,IX14,IX15,NROW,NUBW,
     6ST(360,360),D(600),
     1NLS,LN,IFLAG,IERR,ITRANS,ISTER
      INTEGER TS,RL,CRL,UBW                                                MOD0004
      REAL IX,IY,IZ,L                                                      MOD0005
C
C  INPUT STRUCTURE DATA AND FORM STIFFNESS MATRIX
C
      CALL SDATA                                                           MOD0006
      IF(IFIRST.NE.0) GO TO 1400                                           MOD0007
C
C  CHECK FOR PROPER INPUT OF MEMBER PROPERTIES
C
      CALL TSCHK                                                           MOD0008
      IF(IERR.EQ.1) GO TO 2500                                             MOD0009
C
C  OUTPUT INITIAL MEMBER PROPERTIES AND COST DATA
C
      CALL OUTPUT                                                          MOD0010
      WRITE(6,119)                                                         MOD0011
  119 FORMAT(1X," ")                                                       MOD0012
      WRITE(6,120)                                                         MOD0013
  120 FORMAT(1X,"INITIAL WEIGHT AND COST EVALUATION")                      MOD0014
      WRITE(6,125)                                                         MOD0015
  125 FORMAT(1X,"----------------------------------"//)                    MOD0016
      CALL COSTWT                                                          MOD0017
 1400 CALL STIFF                                                           MOD0018
      LN=0                                                                 MOD0019
      IF(IFIRST.NE.0) GO TO 2001                                           MOD0020
C
C  INPUT LOAD DATA
C
      WRITE(6,1450)                                                        MOD0021
 1450 FORMAT(1H1,"APPLIED LOADS")                                          MOD0022
      WRITE(6,1451)                                                        MOD0023
 1451 FORMAT(1X,"-------------"/)                                          MOD0024
 2000 CALL LDATA                                                           MOD0025
```

```
      IF(LN.LT.NLS) GO TO 2000                                          MOD00026
 2001 IF(IFLAG.EQ.0) GO TO 1501                                         MOD00027
C
C  FROM THIS POINT DOWN TO STATEMENT 1501 STFDES ANALYSIS IS PERFORMED
C
      CALL STFDES                                                       MOD00028
      IF(IERR.EQ.1) GO TO 2500                                          MOD00029
C
C  PERFORM CHOLESKY DECOMPOSITION OF REDUCED STIFFNESS MATRIX
C
      CALL DCBAND(NROW,NUBW,ST,360,+3,+1502)                            MOD00030
 1502 LN=0                                                              MOD00031
 1600 LN=LN+1                                                           MOD00032
      IF(ISTFD.NE.0) GO TO 1601                                         MOD00033
C
C  FORM REDUCED LOAD VECTOR
C
      CALL RLOADV                                                       MOD00034
 1601 LN1=LN+NLS                                                        MOD00035
C
C  READ IN REDUCED LOAD VECTOR
C
      READ(15"LN1) (AC(I),I=1,NROW)                                     MOD00036
C
C  CALCULATE INDEPENDENT DISPLACEMENTS AND DISPLACEMENT MEASURES
C
      CALL SBAND(NROW,NUBW,ST,AC,D,360)                                 MOD00037
C
C  EXPAND REDUCED DISPLACEMENT VECTOR TO FORM VECTOR OF TRUE JOINT DISPL
C                                                              ACEMENTS
C
      CALL EXPRD                                                        MOD00038
C
C  COMPUTE REACTIONS AND MEAS
C
      WRITE(6,1550) LN                                                  MOD00039
      WRITE(6,1551)                                                     MOD00040
      CALL RESULT                                                       MOD00041
C
C  CALCULATE EXACT VALUE OF DISPLACEMENT WHICH IS CRITICAL
C
      CALL DEXACT                                                       MOD00042
      IF(LN.LT.NLS) GO TO 1600                                          MOD00043
      GO TO 2500                                                        MOD00044
C
C  FROM THIS POINT TO THE END OF THIS ROUTINE A REGULAR ANALYSIS IS PERF
C                                                              ORMED
C
```

```
C
C  PERFORM CHOLESKY DECOMPOSITION OF ORIGINAL STIFFNESS MATRIX
C
 1501 CALL DCBAND(N,UBW,S,600,+3,+1503)                                 MOD0045
 1503 LN=0                                                              MOD0046
 1504 LN=LN+1                                                           MOD0047
C
C  READ IN UNMODIFIED LOAD VECTOR
C
      READ(15"LN)(AC(I),I=1,N)                                          MOD0048
C
C  CALCULATE ACTUAL JOINT DISPLACEMENTS
C
      CALL SBAND(N,UBW,S,AC,D,600)                                      MOD0049
C
C  COMPUTE REACTIONS AND MEAS
C
      WRITE(6,1550) LN                                                  MOD0050
 1550 FORMAT(1H1,"DISPLACEMENTS AND MEMBER END ACTIONS FOR LOADING NO.",MOD0051
     1I5)
      WRITE(6,1551)                                                     MOD0052
 1551 FORMAT(1X,"--------------------------------------------------------MOD0053
     1--"//)
      CALL RESULT                                                       MOD0054
C
C  CALCULATE EXACT VALUE OF DISPLACEMENT WHICH IS CRITICAL
C
      CALL DEXACT                                                       MOD0055
      IF(LN.LT.NLS) GO TO 1504                                          MOD0056
      GO TO 2500                                                        MOD0057
    3 PRINT 6                                                           MOD0058
    6 FORMAT(//"DCBAND FAILS")                                          MOD0059
      GO TO 2500                                                        MOD0060
 2500 RETURN                                                            MOD0061
      END                                                               MOD0062
```

```
*DECK SDATA                                                             SDA0000
      SUBROUTINE SDATA                                                  SDA0001
      COMMON DUM5(972),DUM6(39),IDUM3(44),IDUM4(11),                    SDA0002
     1IX16,IX17,IX18,IX19,IX20,IX21,IX22,IX23,IX24,IX25,IX26,IX27,
     2IDUM7(306),
     5COLID1(48),COLID2(48),BMID1(38),BMID2(38),BRID1(20),BRID2(20)
      COMMON ITAB(300),ISEC(300),IFIRST,M,E,DJ(600),                    SDA0003
     1X(100),Y(100),Z(100),L(300),AX(300),IX(300),IY(300),IZ(300),
     2AR(600),A(600),AC(600),AE(600),S(600,600),R(300,9),
     3AML(300,12),SM(12,12),SMD(12,12),SMR(12,12),AM(12),AMD(12),G,
     4JJ(300),JK(300),LML(300),JF,KE,TS,NDJ,NR,
     5RL(600),CRL(600),N,UBW,NJ,IX11,IX12,IX13,IX14,IX15,NROW,NUBW,
     6ST(360,360),D(600),
     1NLS,LN,IFLAG,IERR,ITRANS,ISTEP
      INTEGER TS,RL,CRL,UBW,SN,AA(300)                                  SDA0004
      DIMENSION XPS(300),YPS(300),ZPS(300)                              SDA0005
      REAL IX,IY,IZ,L                                                   SDA0006
C           CONTROL DATA, STRUCTURE PARAMETERS, AND ELASTIC MODULI
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                               CCCCCCCC
C
      IF(IFIRST.NE.0) GO TO 82                                          SDA0007
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                               CCCCCCCC
      WRITE(6,1)                                                        SDA0008
    1 FORMAT(1H1,"ANALYSIS OF FRAMED STRUCTURES FR2"/)                  SDA0009
    5 READ(5,6) SN,TS                                                   SDA0010
    6 FORMAT(2I3)                                                       SDA0011
C
C  ZERO OUT RESTRAINT LIST AND ROTATION MATRIX
C
      DO 8 I=1,600                                                      SDA0012
      RL(I)=0                                                           SDA0013
 8    CONTINUE                                                          SDA0014
      DO 200 I=1,300                                                    SDA0015
      AA(I)=0                                                           SDA0016
      DO 200 J=1,9                                                      SDA0017
      R(I,J)=0.0                                                        SDA0018
  200 CONTINUE                                                          SDA0019
 10   IF(TS.NE.1)GO TO 15                                               SDA0020
      WRITE(6,13) SN                                                    SDA0021
   13 FORMAT(1X,"STRUCTURE NO.",I5,2X,"CONTINUOUS BEAM"/)               SDA0022
      READ(5,14)M,NR,NRJ,E                                              SDA0023
 14   FORMAT(3I3,F7.0)                                                  SDA0024
      NJ=M+1                                                            SDA0025
      NDJ=2                                                             SDA0026
      GO TO 40                                                          SDA0027
```

```
15    IF(TS.NE.2)GO TO 20                                              SDA0028
      WRITE(6,18) SN                                                   SDA0029
  18  FORMAT(1X,"STRUCTURE NO.",I5,2X,"PLANE TRUSS"/)                  SDA0030
      READ(5,19)M,NJ,NR,NRJ,E                                          SDA0031
19    FORMAT(4I3,F7.0)                                                 SDA0032
      NDJ=2                                                            SDA0033
      GO TO 40                                                         SDA0034
20    IF(TS.NE.3)GO TO 25                                              SDA0035
      WRITE(6,21) SN                                                   SDA0036
  21  FORMAT(1X,"STRUCTURE NO.",I5,2X,"PLANE FRAME"/)                  SDA0037
22    READ(5,19)M,NJ,NR,NRJ,E                                          SDA0038
      NDJ=3                                                            SDA0039
      GO TO 40                                                         SDA0040
25    IF(TS.NE.4)GO TO 30                                              SDA0041
      WRITE(6,26) SN                                                   SDA0042
  26  FORMAT(1X,"STRUCTURE NO.",I5,2X,"GRID"/)                         SDA0043
      READ(5,27)M,NJ,NR,NRJ,E,G                                        SDA0044
27    FORMAT(4I3,2F7.0)                                                SDA0045
      NDJ=3                                                            SDA0046
      GO TO 40                                                         SDA0047
30    IF(TS.NE.5)GO TO 35                                              SDA0048
      WRITE(6,31) SN                                                   SDA0049
  31  FORMAT(1X,"STRUCTURE NO.",I5,2X,"SPACE TRUSS"/)                  SDA0050
      GO TO 22                                                         SDA0051
  35  WRITE(6,32) SN                                                   SDA0052
  32  FORMAT(1X,"STRUCTURE NO.",I5,2X,"SPACE FRAME"/)                  SDA0053
      READ(5,27)M,NJ,NR,NRJ,E,G                                        SDA0054
      NDJ=6                                                            SDA0055
40    PRINT 41                                                         SDA0056
41    FORMAT("0STRUCTURE DATA")                                        SDA0057
      PRINT 42                                                         SDA0058
42    FORMAT("    M     N   NJ   NR   NRJ       E          G")         SDA0059
      N=(NDJ*NJ)-NR                                                    SDA0060
      IF(TS.EQ.4.OR.TS.EQ.6)GO TO 45                                   SDA0061
      WRITE(6,43)M,N,NJ,NR,NRJ,E                                       SDA0062
43    FORMAT(" ",I3,4I4,F9.1)                                          SDA0063
      GO TO 47                                                         SDA0064
45    WRITE(6,46)M,N,NJ,NR,NRJ,E,G                                     SDA0065
46    FORMAT(" ",I3,4I4,2F9.1)                                         SDA0066
C          JOINT COORDINATES
47    IF(TS.EQ.1)GO TO 60                                              SDA0067
      PRINT 48                                                         SDA0068
48    FORMAT("0COORDINATES OF JOINTS")                                 SDA0069
      PRINT 49                                                         SDA0070
49    FORMAT(" JOINT    X         Y         Z")                        SDA0071
      IF(TS.GE.5)GO TO 55                                              SDA0072
      DO 52 K=1,NJ                                                     SDA0073
      READ(5,50) J, X(J), Y(J)                                         SDA0074
```

```
 50    FORMAT(I3,2F7.0)                                                 SDA0075
       WRITE(6,51)J,X(J),Y(J)                                           SDA0076
 51    FORMAT(" ",I3,2F8.2)                                             SDA0077
 52    CONTINUE                                                         SDA0078
       GO TO 60                                                         SDA0079
 55    DO 57 K=1,NJ                                                     SDA0080
       READ(5,58) J, X(J), Y(J), Z(J)                                   SDA0081
 58    FORMAT(I3,3F7.0)                                                 SDA0082
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                               CCCCCCCC
C
       WRITE(20"J) X(J),Y(J),Z(J)                                       SDA0083
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                               CCCCCCCC
C
       WRITE(6,56)J,X(J),Y(J),Z(J)                                      SDA0084
 56    FORMAT(" ",I3,3F8.2)                                             SDA0085
 57    CONTINUE                                                         SDA0086
C         MEMBER INFORMATION, ROTATION MATRIX
 60    PRINT 61                                                         SDA0087
 61    FORMAT("0MEMBER INFORMATION")                                    SDA0088
       PRINT 62                                                         SDA0089
   62  FORMAT(1X,"MEMBER    JJ    JK   AA TAB SEC")                     SDA0090
       IF(TS.EQ.1)GO TO 64                                              SDA0091
       IF(TS.LT.5)GO TO 70                                              SDA0092
       GO TO 82                                                         SDA0093
 64    DO 66 K=1,M                                                      SDA0094
       READ(5,67) I, L(I), IZ(I)                                        SDA0095
 67    FORMAT(I3,2F7.0)                                                 SDA0096
       JJ(I)=I                                                          SDA0097
       JK(I)=I+1                                                        SDA0098
       WRITE(6,65)I,JJ(I),JK(I),IZ(I),L(I)                              SDA0099
 65    FORMAT(" ",I4,I5,I3,F31.2,F10.2)                                 SDA0100
 66    CONTINUE                                                         SDA0101
       GO TO 100                                                        SDA0102
 70    DO 81 K=1,M                                                      SDA0103
       IF(TS.EQ.2)GO TO 71                                              SDA0104
       IF(TS.EQ.3)GO TO 72                                              SDA0105
       READ(5,68)I,JJ(I),JK(I),IX(I),IY(I)                              SDA0106
 68    FORMAT(3I3,2F7.0)                                                SDA0107
       GO TO 73                                                         SDA0108
 71    READ(5,14)I,JJ(I),JK(I),AX(I)                                    SDA0109
       GO TO 73                                                         SDA0110
 72    READ(5,68)I,JJ(I),JK(I),AX(I),IZ(I)                              SDA0111
 73    IF(JJ(I).LE.JK(I))GO TO 74                                       SDA0112
       J=JJ(I)                                                          SDA0113
```

```
      JJ(I)=JK(I)                                                       SDA0114
      JK(I)=J                                                           SDA0115
   74 JAC1=JJ(I)                                                        SDA0116
      JAC2=JK(I)                                                        SDA0117
      XCL=X(JAC2)-X(JAC1)                                               SDA0118
      YCL=Y(JAC2)-Y(JAC1)                                               SDA0119
      L(I)=SQRT((XCL**2)+(YCL**2))                                      SDA0120
      CX=XCL/L(I)                                                       SDA0121
      CY=YCL/L(I)                                                       SDA0122
      IF(TS.EQ.2)GO TO 78                                               SDA0123
      IF(TS.EQ.3)GO TO 76                                               SDA0124
      WRITE(6,75)I,JJ(I),JK(I),IX(I),IY(I),L(I)                         SDA0125
   75 FORMAT(" ",I4,I5,I3,F15.2,F8.2,F18.2)                             SDA0126
      GO TO 80                                                          SDA0127
   76 WRITE(6,77)I,JJ(I),JK(I),AX(I),IZ(I),L(I)                         SDA0128
   77 FORMAT(" ",I4,I5,I3,F7.2,F24.2,F10.2)                             SDA0129
      GO TO 80                                                          SDA0130
   78 WRITE(6,79)I,JJ(I),JK(I),AX(I),L(I)                               SDA0131
   79 FORMAT(" ",I4,I5,I3,F7.2,F34.2)                                   SDA0132
      R(I,1)=CX                                                         SDA0133
      R(I,4)=CX                                                         SDA0134
      R(I,2)=CY                                                         SDA0135
      R(I,3)=-CY                                                        SDA0136
      GO TO 81                                                          SDA0137
   80 R(I,3)=0.0                                                        SDA0138
      R(I,6)=0.0                                                        SDA0139
      R(I,7)=0.0                                                        SDA0140
      R(I,8)=0.0                                                        SDA0141
      R(I,1)=CX                                                         SDA0142
      R(I,5)=CX                                                         SDA0143
      R(I,9)=1.0                                                        SDA0144
      R(I,2)=CY                                                         SDA0145
      R(I,4)=-CY                                                        SDA0146
   81 CONTINUE                                                          SDA0147
      GO TO 100                                                         SDA0148
   82 DO 99 K=1,M                                                       SDA0149
      IF(TS.EQ.5) GO TO 83                                              SDA0150
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                CCCCCCCC
C
      IF(IFIRST.EQ.0) GO TO 869                                         SDA0151
      I=K                                                               SDA0152
      GO TO 871                                                         SDA0153
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                CCCCCCCC
C
C  READ IN MEMBER AND ITS CORRESPONDING JOINTS AS WELL AS TABLE AND SECT
```

```
                                                                  ION NO.
C
C  READ IN FROM DISK SECTION PROPERTIES ASSOCIATED WITH TABLE AND SECTIO
C                                                                   N
C
C
  869 READ(5,87) I,JJ(I),JK(I),AA(I),ITAB(I),ISEC(I),RHO,U            SDA0154
   87 FORMAT(6I5,2F10.2)                                               SDA0155
  871 ITABI=ITAB(I)                                                    SDA0156
      ISECI=ISEC(I)                                                    SDA0157
      READ(ITABI"ISECI) AX(I),IX(I),IY(I),IZ(I)                        SDA0158
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                               CCCCCCCC
      GO TO 84                                                         SDA0159
 83   READ(5,14)I,JJ(I),JK(I),AX(I)                                    SDA0160
 84   IF(JJ(I).LE.JK(I))GO TO 85                                       SDA0161
      J=JJ(I)                                                          SDA0162
      JJ(I)=JK(I)                                                      SDA0163
      JK(I)=J                                                          SDA0164
  85  JAC1=JJ(I)                                                       SDA0165
      JAC2=JK(I)                                                       SDA0166
      XCL=X(JAC2)-X(JAC1)                                              SDA0167
      YCL=Y(JAC2)-Y(JAC1)                                              SDA0168
      ZCL=Z(JAC2)-Z(JAC1)                                              SDA0169
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                               CCCCCCCC
C
      IF(IFIRST.NE.0) GO TO 851                                        SDA0170
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                               CCCCCCCC
C
      L(I)=SQRT((XCL**2)+(YCL**2)+(ZCL**2))                            SDA0171
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                               CCCCCCCC
C
C  WRITE RHO U AND LENGTH OUT TO DISK
C
      WRITE(19"I) RHO,U,L(I)                                           SDA0172
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                               CCCCCCCC
C
  851 CX=XCL/L(I)                                                      SDA0173
      CY=YCL/L(I)                                                      SDA0174
      CZ=ZCL/L(I)                                                      SDA0175
      IF(TS.EQ.5)GO TO 90                                              SDA0176
C
```

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                               CCCCCCCC
C
      IF(IFIRST.NE.0) GO TO 92                                          SDA00177
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                               CCCCCCCC
C
      WRITE(6,86) I,JJ(I),JK(I),AA(I),ITAP(I),ISEC(I)                   SDA00178
   86 FORMAT(1X,I4,I7,I6,3I4)                                           SDA00179
      IF(AA(I).NE.1) GO TO 92                                           SDA00180
      READ(5,58)I,XP,YP,ZP                                              SDA00181
      JJTEMP=JJ(I)                                                      SDA00182
      XPS(I)=XP-X(JJTEMP)                                               SDA00183
      YPS(I)=YP-Y(JJTEMP)                                               SDA00184
      ZPS(I)=ZP-Z(JJTEMP)                                               SDA00185
      GO TO 92                                                          SDA00186
 90   WRITE(6,91)I,JJ(I),JK(I),AX(I),L(I)                               SDA00187
 91   FORMAT(" ",I4,I5,I3,F7.2,F34.2)                                   SDA00188
 92   Q=SQRT((CX**2)+(CZ**2))                                           SDA00189
      IF(Q.GE.0.001)GO TO 95                                            SDA00190
      R(I,1)=0.0                                                        SDA00191
      R(I,3)=0.0                                                        SDA00192
      R(I,5)=0.0                                                        SDA00193
      R(I,6)=0.0                                                        SDA00194
      R(I,7)=0.0                                                        SDA00195
      R(I,8)=0.0                                                        SDA00196
      R(I,2)=CY                                                         SDA00197
      R(I,4)=-CY                                                        SDA00198
      R(I,9)=1.0                                                        SDA00199
      IF(IS.EQ.6.AND.AA(I).EQ.1) GO TO 93                               SDA00200
      GO TO 99                                                          SDA00201
   93 SQ=SQRT((XPS(I)**2)+(ZPS(I)**2))                                  SDA00202
      COSA=-XPS(I)*CY/SQ                                                SDA00203
      SINA=ZPS(I)/SQ                                                    SDA00204
      R(I,4)=-CY*COSA                                                   SDA00205
      R(I,6)=SINA                                                       SDA00206
      R(I,7)=CY*SINA                                                    SDA00207
      R(I,9)=COSA                                                       SDA00208
      GO TO 99                                                          SDA00209
 95   R(I,1)=CX                                                         SDA00210
      R(I,2)=CY                                                         SDA00211
      R(I,3)=CZ                                                         SDA00212
      R(I,4)=-CX*CY/Q                                                   SDA00213
      R(I,5)=Q                                                          SDA00214
      R(I,6)=-CY*CZ/Q                                                   SDA00215
      R(I,7)=-CZ/Q                                                      SDA00216
      R(I,8)=0.0                                                        SDA00217
```

```
      R(I,9)=CX/Q                                                       SDA0218
      IF(TS.EQ.6.AND.AA(I).EQ.1) GO TO 96                               SDA0219
      GO TO 99                                                          SDA0220
   96 YPG=R(I,4)*XPS(I)+R(I,5)*YPS(I)+R(I,6)*ZPS(I)                     SDA0221
      ZPG=R(I,7)*XPS(I)+R(I,8)*YPS(I)+R(I,9)*ZPS(I)                     SDA0222
      SQ=SQRT((YPG**2)+(ZPG**2))                                        SDA0223
      COSA=YPG/SQ                                                       SDA0224
      SINA=ZPG/SQ                                                       SDA0225
      R(I,4)=(-CY*CY*COSA-CZ*SINA)/Q                                    SDA0226
      R(I,5)=Q*COSA                                                     SDA0227
      R(I,8)=-Q*SINA                                                    SDA0228
      R(I,6)=(-CY*CZ*COSA+CX*SINA)/Q                                    SDA0229
      R(I,7)=(CX*CY*SINA-CZ*COSA)/Q                                     SDA0230
      R(I,9)=(CY*CZ*SINA+CX*COSA)/Q                                     SDA0231
      GO TO 99                                                          SDA0232
   99 CONTINUE                                                          SDA0233
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                               CCCCCCCC
C
      IF(IFIRST.NE.0) GO TO 117                                         SDA0234
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                               CCCCCCCC
C        JOINT RESTRAINT LIST, CUMULATIVE RESTRAINT LIST
  100 PRINT 101                                                         SDA0235
  101 FORMAT("0JOINT RESTRAINTS")                                       SDA0236
      PRINT 102                                                         SDA0237
  102 FORMAT(" JOINT RL1 RL2 RL3 RL4 RL5 RL6")                          SDA0238
      IF(NRJ.EQ.0)GO TO 115                                             SDA0239
      IF(TS.GE.3)GO TO 105                                              SDA0240
      DO 104 J=1,NRJ                                                    SDA0241
      READ(5,107)K,RL(2*K-1),RL(2*K)                                    SDA0242
  107 FORMAT(3I3)                                                       SDA0243
      WRITE(6,103)K,RL(2*K-1),RL(2*K)                                   SDA0244
  103 FORMAT(" ",I3,I5,I4)                                              SDA0245
  104 CONTINUE                                                          SDA0246
      GO TO 115                                                         SDA0247
  105 IF(TS.EQ.6)GO TO 110                                              SDA0248
      DO 109 J=1,NRJ                                                    SDA0249
      READ(5,108)K,RL(3*K-2),RL(3*K-1),RL(3*K)                          SDA0250
  108 FORMAT(4I3)                                                       SDA0251
      WRITE(6,106) K,RL(3*K-2),RL(3*K-1),RL(3*K)                        SDA0252
  106 FORMAT(" ",I3,I5,2I4)                                             SDA0253
  109 CONTINUE                                                          SDA0254
      GO TO 115                                                         SDA0255
  110 DO 114 J=1,NRJ                                                    SDA0256
      READ(5,112)K,RL(6*K-5),RL(6*K-4),RL(6*K-3),RL(6*K-2),RL(6*K-1),   SDA0257
     1RL(6*K)
```

```
112  FORMAT(7I3)
     WRITE(6,111)K,RL(6*K-5),RL(6*K-4),RL(6*K-3),RL(6*K-2),RL(6*K-1),
    1RL(6*K)
111  FORMAT (" ",I3,I5,5I4)
114  CONTINUE
115  CRL(1)=RL(1)
     NPNR=N+NR
     DO 116 K=2,NPNR
     CRL(K)=CRL(K-1)+RL(K)
116  CONTINUE
 117 RETURN
     END
```

```
*DECK STIFF
      SUBROUTINE STIFF
      COMMON DUM5(972),DUM6(39),IDUM3(44),IDUM4(11),
     1IY16,IX17,IX18,IX19,IX20,IX21,IX22,IX23,IX24,IX25,IX26,IX27,
     2IDUM7(306),
     5COLID1(48),COLID2(48),BMID1(38),BMID2(38),BRID1(20),BRID2(20)
      COMMON ITAB(300),ISEC(300),IFIRST,M,E,DJ(600),
     1X(100),Y(100),Z(100),L(300),AX(300),IX(300),IY(300),IZ(300),
     2AR(600),A(600),AC(600),AE(600),S(600,600),P(300,9),
     3AML(300,12),SM(12,12),SMD(12,12),SMR(12,12),AM(12),AMD(12),G,
     4JJ(300),JK(300),LML(300),JF,KE,TS,NDJ,NP,
     5PL(600),CRL(600),N,UBW,NJ,IX11,IX12,IX13,IX14,IX15,NROW,NUBW,
     6ST(360,360),D(600),
     1NLS,LN,IFLAG,IEPR,ITRANS,ISTFD
      INTEGER TS,RL,CRL,UBW
      INTEGER RCHK,C,ROW,COL,DOF,DD1,DD2,DD3,DD4,DD5,DD6
      REAL IX,IY,IZ,L
      DO 200 I=1,600
      DO 200 J=1,600
      S(I,J)=0.0
  200 CONTINUE
      DO 201 I=1,12
      DO 201 J=1,12
      SM(I,J)=0.0
      SMR(I,J)=0.0
      SMD(I,J)=0.0
  201 CONTINUE
      UBW=0
      I=0
    2 I=I+1
      IF(I.GT.M) GO TO 87
      IF(TS.LT.3) GO TO 4
      IF(TS.LT.6) GO TO 5
      J1=6*JJ(I)-5
      J2=6*JJ(I)-4
      J3=6*JJ(I)-3
      J4=6*JJ(I)-2
      J5=6*JJ(I)-1
      J6=6*JJ(I)
      K1=6*JK(I)-5
      K2=6*JK(I)-4
      K3=6*JK(I)-3
      K4=6*JK(I)-2
      K5=6*JK(I)-1
      K6=6*JK(I)
      GO TO 7
    4 J1=2*JJ(I)-1
      J2=2*JJ(I)
```

```
      K1=2*JK(I)-1
      K2=2*JK(I)
      GO TO 7
    5 J1=3*JJ(I)-2
      J2=3*JJ(I)-1
      J3=3*JJ(I)
      K1=3*JK(I)-2
      K2=3*JK(I)-1
      K3=3*JK(I)
      GO TO 7
    7 IF(TS.EQ.1.OR.TS.EQ.3.OR.TS.EQ.6) GO TO 10
    8 IF(TS.EQ.2.OR.TS.EQ.3.OR.TS.EQ.5.OR.TS.EQ.6) GO TO 11
    9 IF(TS.EQ.4.OR.TS.EQ.6) GO TO 12
      GO TO 15
   10 SCM2Z=(4.0*E*IZ(I))/L(I)
      SCM3Z=(1.5*SCM2Z)/L(I)
      SCM4Z=(2.0*SCM3Z)/L(I)
      GO TO 8
   11 SCM1A=(E*AX(I))/L(I)
      GO TO 9
   12 SCM1B=(G*IX(I))/L(I)
      SCM2Y=(4.0*E*IY(I))/L(I)
      SCM3Y=(1.5*SCM2Y)/L(I)
      SCM4Y=(2.0*SCM3Y)/L(I)
      GO TO 15
   15 IF(TS.EQ.1) GO TO 16
      IF(TS.EQ.2) GO TO 17
      IF(TS.EQ.3) GO TO 18
      IF(TS.EQ.4) GO TO 19
      IF(TS.EQ.5) GO TO 20
      IF(TS.EQ.6) GO TO 21
   16 SMD(1,1)=SCM4Z
      SMD(3,3)=SCM4Z
      SMD(1,3)=-SCM4Z
      SMD(1,2)=SCM3Z
      SMD(2,1)=SCM3Z
      SMD(3,1)=-SCM4Z
      SMD(1,4)=SCM3Z
      SMD(4,1)=SCM3Z
      SMD(2,3)=-SCM3Z
      SMD(3,2)=-SCM3Z
      SMD(3,4)=-SCM3Z
      SMD(4,3)=-SCM3Z
      SMD(2,2)=SCM2Z
      SMD(4,4)=SCM2Z
      SMD(2,4)=SCM2Z/2.0
      SMD(4,2)=SCM2Z/2.0
      GO TO 26
```

```
17 SM(1,1)=SCM1A
   SM(3,3)=SCM1A
   SM(1,3)=-SCM1A
   SM(3,1)=-SCM1A
   GO TO 22
18 SM(1,1)=SCM1A
   SM(4,4)=SCM1A
   SM(1,4)=-SCM1A
   SM(4,1)=-SCM1A
   SM(2,2)=SCM4Z
   SM(5,5)=SCM4Z
   SM(2,5)=-SCM4Z
   SM(5,2)=-SCM4Z
   SM(2,3)=SCM3Z
   SM(3,2)=SCM3Z
   SM(2,6)=SCM3Z
   SM(6,2)=SCM3Z
   SM(3,5)=-SCM3Z
   SM(5,3)=-SCM3Z
   SM(5,6)=-SCM3Z
   SM(6,5)=-SCM3Z
   SM(3,3)=SCM2Z
   SM(6,6)=SCM2Z
   SM(6,3)=SCM2Z/2.0
   SM(3,6)=SCM2Z/2.0
   GO TO 24
19 SM(1,1)=SCM1B
   SM(4,4)=SCM1B
   SM(4,1)=-SCM1B
   SM(1,4)=-SCM1B
   SM(2,2)=SCM2Y
   SM(5,5)=SCM2Y
   SM(2,5)=SCM2Y/2.0
   SM(5,2)=SCM2Y/2.0
   SM(2,6)=SCM3Y
   SM(6,2)=SCM3Y
   SM(5,6)=SCM3Y
   SM(6,5)=SCM3Y
   SM(2,3)=-SCM3Y
   SM(3,2)=-SCM3Y
   SM(3,5)=-SCM3Y
   SM(5,3)=-SCM3Y
   SM(3,3)=SCM4Y
   SM(6,6)=SCM4Y
   SM(3,6)=-SCM4Y
   SM(6,3)=-SCM4Y
   GO TO 24
20 SM(1,1)=SCM1A
```

```
      SM(4,4)=SCM1A
      SM(1,4)=-SCM1A
      SM(4,1)=-SCM1A
      GO TO 24
   21 SM(1,1)=SCM1A
      SM(7,7)=SCM1A
      SM(1,7)=-SCM1A
      SM(7,1)=-SCM1A
      SM(2,2)=SCM4Z
      SM(8,8)=SCM4Z
      SM(2,6)=SCM3Z
      SM(6,2)=SCM3Z
      SM(2,12)=SCM3Z
      SM(12,2)=SCM3Z
      SM(6,8)=-SCM3Z
      SM(8,6)=-SCM3Z
      SM(8,12)=-SCM3Z
      SM(12,8)=-SCM3Z
      SM(3,3)=SCM4Y
      SM(9,9)=SCM4Y
      SM(3,5)=-SCM3Y
      SM(5,3)=-SCM3Y
      SM(3,11)=-SCM3Y
      SM(11,3)=-SCM3Y
      SM(2,8)=-SCM4Z
      SM(8,2)=-SCM4Z
      SM(3,9)=-SCM4Y
      SM(9,3)=-SCM4Y
      SM(4,4)=SCM1B
      SM(10,10)=SCM1B
      SM(4,10)=-SCM1B
      SM(10,4)=-SCM1B
      SM(5,5)=SCM2Y
      SM(11,11)=SCM2Y
      SM(5,11)=SCM2Y/2.0
      SM(11,5)=SCM2Y/2.0
      SM(5,9)=SCM3Y
      SM(9,5)=SCM3Y
      SM(9,11)=SCM3Y
      SM(11,9)=SCM3Y
      SM(6,6)=SCM2Z
      SM(12,12)=SCM2Z
      SM(6,12)=SCM2Z/2.0
      SM(12,6)=SCM2Z/2.0
      GO TO 24
   22 DO 221 K=1,2
      DO 221 J=1,4
      SMR(J,2*K-1)=SM(J,2*K-1)*R(I,1)+SM(J,2*K)*R(I,3)
```

```
      SMR(J,2*K)=SM(J,2*K-1)*R(I,2)+SM(J,2*K)*R(I,4)
  221 CONTINUE
      DO 222 J=1,2
      DO 222 K=1,4
      SMD(2*J-1,K)=R(I,1)*SMR(2*J-1,K)+R(I,3)*SMR(2*J,K)
      SMD(2*J,K)=R(I,2)*SMR(2*J-1,K)+R(I,4)*SMR(2*J,K)
  222 CONTINUE
      GO TO 26
   24 KL=2*NDJ/3
      JL=2*NDJ
      DO 241 K=1,KL
      DO 241 J=1,JL
      SMR(J,3*K-2)=SM(J,3*K-2)*R(I,1)+SM(J,3*K-1)*R(I,4)+SM(J,3*K)*R(I
     1,7)
      SMR(J,3*K-1)=SM(J,3*K-2)*R(I,2)+SM(J,3*K-1)*R(I,5)+SM(J,3*K)*R(I
     1,8)
      SMR(J,3*K)=SM(J,3*K-2)*R(I,3)+SM(J,3*K-1)*R(I,6)+SM(J,3*K)*R(I,9)
  241 CONTINUE
      DO 251 J=1,KL
      DO 251 K=1,JL
      SMD(3*J-2,K)=R(I,1)*SMR(3*J-2,K)+R(I,4)*SMR(3*J-1,K)+R(I,7)*SMR(
     13*J,K)
      SMD(3*J-1,K)=R(I,2)*SMR(3*J-2,K)+R(I,5)*SMR(3*J-1,K)+R(I,8)*SMR(
     13*J,K)
      SMD(3*J,K)=R(I,3)*SMR(3*J-2,K)+R(I,6)*SMR(3*J-1,K)+R(I,9)*SMR(3*J
     1,K)
  251 CONTINUE
      IF(TS.LT.6) GO TO 39
      GO TO 40
   26 IF(RL(J1).NE.0) GO TO 31
      ROW=J1-CRL(J1)
      S(ROW,1)=S(ROW,1)+SMD(1,1)
      IF(RL(J2).NE.0)    GO TO 28
      S(ROW,2)=S(ROW,2)+SMD(1,2)
   28 IF(RL(K1).NE.0   ) GO TO 29
      COL=K1-CRL(K1)-ROW+1
      S(ROW,COL)=SMD(1,3)
   29 IF(RL(K2).NE.0)    GO TO 30
      COL=K2-CRL(K2)-ROW+1
      S(ROW,COL)=SMD(1,4)
   30 IF(COL.GT.UBW) UBW=COL
   31 IF(RL(J2).NE.0)    GO TO 35
      ROW=J2-CRL(J2)
      S(ROW,1)=S(ROW,1)+SMD(2,2)
      IF(RL(K1).NE.0)    GO TO 33
      COL=K1-CRL(K1)-ROW+1
      S(ROW,COL)=SMD(2,3)
   33 IF(RL(K2).NE.0)    GO TO 34
```

```
      COL=K2-CRL(K2)-ROW+1
      S(ROW,COL)=SMD(2,4)
   34 IF(COL.GT.UBW) UBW=COL
   35 IF(RL(K1).NE.0)    GO TO 37
      ROW=K1-CRL(K1)
      S(ROW,1)=S(ROW,1)+SMD(3,3)
      IF(RL(K2).NE.0)    GO TO 37
      S(ROW,2)=S(ROW,2)+SMD(3,4)
   37 IF(RL(K2).NE.0) GO TO 2
      ROW=K2-CRL(K2)
      S(ROW,1)=S(ROW,1)+SMD(4,4)
      GO TO 2
   39 IF(RL(J1).NE.0) GO TO 391
      ROW=J1-CRL(J1)
      S(ROW,1)=S(ROW,1)+SMD(1,1)
      IF(RL(J2).NE.0)    GO TO 3902
      S(ROW,2)=S(ROW,2)+SMD(1,2)
 3902 IF(RL(J3).NE.0)    GO TO 3903
      COL=J3-CRL(J3)-ROW+1
      S(ROW,COL)=S(ROW,COL)+SMD(1,3)
 3903 IF(RL(K1).NE.0)    GO TO 3904
      COL=K1-CRL(K1)-ROW+1
      S(ROW,COL)=SMD(1,4)
 3904 IF(RL(K2).NE.0)    GO TO 3905
      COL=K2-CRL(K2)-ROW+1
      S(ROW,COL)=SMD(1,5)
 3905 IF(RL(K3).NE.0)    GO TO 3906
      COL=K3-CRL(K3)-ROW+1
      S(ROW,COL)=SMD(1,6)
 3906 IF(COL.GT.UBW) UBW=COL
  391 IF(RL(J2).NE.0)    GO TO 392
      ROW=J2-CRL(J2)
      S(ROW,1)=S(ROW,1)+SMD(2,2)
      IF(RL(J3).NE.0)    GO TO 3912
      S(ROW,2)=S(ROW,2)+SMD(2,3)
 3912 IF(RL(K1).NE.0)    GO TO 3913
      COL=K1-CRL(K1)-ROW+1
      S(ROW,COL)=SMD(2,4)
 3913 IF(RL(K2).NE.0)    GO TO 3914
      COL=K2-CRL(K2)-ROW+1
      S(ROW,COL)=SMD(2,5)
 3914 IF(RL(K3).NE.0)    GO TO 3915
      COL=K3-CRL(K3)-ROW+1
      S(ROW,COL)=SMD(2,6)
 3915 IF(COL.GT.UBW) UBW=COL
  392 IF(RL(J3).NE.0)    GO TO 393
      ROW=J3-CRL(J3)
      S(ROW,1)=S(ROW,1)+SMD(3,3)
```

```
      IF(RL(K1).NE.0)    GO TO 3922
      COL=K1-CRL(K1)-ROW+1
      S(ROW,COL)=SMD(3,4)
 3922 IF(RL(K2).NE.0)    GO TO 3923
      COL=K2-CRL(K2)-ROW+1
      S(ROW,COL)=SMD(3,5)
 3923 IF(RL(K3).NE.0)    GO TO 3924
      COL=K3-CRL(K3)-ROW+1
      S(ROW,COL)=SMD(3,6)
 3924 IF(COL.GT.UBW) UBW=COL
  393 IF(RL(K1).NE.0)    GO TO 3933
      ROW=K1-CRL(K1)
      S(ROW,1)=S(ROW,1)+SMD(4,4)
      IF(RL(K2).NE.0)    GO TO 3932
      S(ROW,2)=S(ROW,2)+SMD(4,5)
 3932 IF(RL(K3).NE.0)    GO TO 3933
      COL=K3-CRL(K3)-ROW+1
      S(ROW,COL)=S(ROW,COL)+SMD(4,6)
 3933 IF(RL(K2).NE.0)    GO TO 3935
      ROW=K2-CRL(K2)
      S(ROW,1)=S(ROW,1)+SMD(5,5)
      IF(RL(K3).NE.0)    GO TO 3935
      S(ROW,2)=S(ROW,2)+SMD(5,6)
 3935 IF(RL(K3).NE.0) GO TO 2
      ROW=K3-CRL(K3)
      S(ROW,1)=S(ROW,1)+SMD(6,6)
      GO TO 2
   40 IF(RL(J1).NE.0) GO TO 4050
      ROW=J1-CRL(J1)
      S(ROW,1)=S(ROW,1)+SMD(1,1)
      IF(RL(J2).NE.0)    GO TO 4001
      S(ROW,2)=S(ROW,2)+SMD(1,2)
 4001 IF(RL(J3).NE.0)    GO TO 4002
      COL=J3-CRL(J3)-ROW+1
      S(ROW,COL)=S(ROW,COL)+SMD(1,3)
 4002 IF(RL(J4).NE.0)    GO TO 4003
      COL=J4-CRL(J4)-ROW+1
      S(ROW,COL)=S(ROW,COL)+SMD(1,4)
 4003 IF(RL(J5).NE.0)    GO TO 4004
      COL=J5-CRL(J5)-ROW+1
      S(ROW,COL)=S(ROW,COL)+SMD(1,5)
 4004 IF(RL(J6).NE.0)    GO TO 4005
      COL=J6-CRL(J6)-ROW+1
      S(ROW,COL)=S(ROW,COL)+SMD(1,6)
 4005 IF(RL(K1).NE.0)    GO TO 4006
      COL=K1-CRL(K1)-ROW+1
      S(ROW,COL)=SMD(1,7)
 4006 IF(RL(K2).NE.0)    GO TO 4007
```

```
      COL=K2-CRL(K2)-ROW+1
      S(ROW,COL)=SMD(1,8)
 4007 IF(RL(K3).NE.0)   GO TO 4008
      COL=K3-CRL(K3)-ROW+1
      S(ROW,COL)=SMD(1,9)
 4008 IF(RL(K4).NE.0)  GO TO 4009
      COL=K4-CRL(K4)-ROW+1
      S(ROW,COL)=SMD(1,10)
 4009 IF(RL(K5).NE.0)   GO TO 4010
      COL=K5-CRL(K5)-ROW+1
      S(ROW,COL)=SMD(1,11)
 4010 IF(RL(K6).NE.0)   GO TO 4011
      COL=K6-CRL(K6)-ROW+1
      S(ROW,COL)=SMD(1,12)
 4011 IF(COL.GT.UBW) UBW=COL
 4050 IF(RL(J2).NE.0)  GO TO 4100
      ROW=J2-CRL(J2)
      S(ROW,1)=S(ROW,1)+SMD(2,2)
      IF(RL(J3).NE.0)   GO TO 4051
      S(ROW,2)=S(ROW,2)+SMD(2,3)
 4051 IF(RL(J4).NE.0)  GO TO 4052
      COL=J4-CRL(J4)-ROW+1
      S(ROW,COL)=S(ROW,COL)+SMD(2,4)
 4052 IF(RL(J5).NE.0)  GO TO 4053
      COL=J5-CRL(J5)-ROW+1
      S(ROW,COL)=S(ROW,COL)+SMD(2,5)
 4053 IF(RL(J6).NE.0)  GO TO 4054
      COL=J6-CRL(J6)-ROW+1
      S(ROW,COL)=S(ROW,COL)+SMD(2,6)
 4054 IF(RL(K1).NE.0)  GO TO 4055
      COL=K1-CRL(K1)-ROW+1
      S(ROW,COL)=SMD(2,7)
 4055 IF(RL(K2).NE.0)  GO TO 4056
      COL=K2-CRL(K2)-ROW+1
      S(ROW,COL)=SMD(2,8)
 4056 IF(RL(K3).NE.0)  GO TO 4057
      COL=K3-CRL(K3)-ROW+1
      S(ROW,COL)=SMD(2,9)
 4057 IF(RL(K4).NE.0)  GO TO 4058
      COL=K4-CRL(K4)-ROW+1
      S(ROW,COL)=SMD(2,10)
 4058 IF(RL(K5).NE.0)  GO TO 4059
      COL=K5-CRL(K5)-ROW+1
      S(ROW,COL)=SMD(2,11)
 4059 IF(RL(K6).NE.0)  GO TO 4060
      COL=K6-CRL(K6)-ROW+1
      S(ROW,COL)=SMD(2,12)
 4060 IF(COL.GT.UBW) UBW=COL
```

```
4100 IF(RL(J3).NE.0)   GO TO 4150
     ROW=J3-CRL(J3)
     S(ROW,1)=S(ROW,1)+SMD(3,3)
     IF(RL(J4).NE.0)   GO TO 4101
     S(ROW,2)=S(ROW,2)+SMD(3,4)
4101 IF(RL(J5).NE.0)   GO TO 4102
     COL=J5-CRL(J5)-ROW+1
     S(ROW,COL)=S(ROW,COL)+SMD(3,5)
4102 IF(RL(J6).NE.0)   GO TO 4103
     COL=J6-CRL(J6)-ROW+1
     S(ROW,COL)=S(ROW,COL)+SMD(3,6)
4103 IF(RL(K1).NE.0)   GO TO 4104
     COL=K1-CRL(K1)-ROW+1
     S(ROW,COL)=SMD(3,7)
4104 IF(RL(K2).NE.0)   GO TO 4105
     COL=K2-CRL(K2)-ROW+1
     S(ROW,COL)=SMD(3,8)
4105 IF(RL(K3).NE.0)   GO TO 4106
     COL=K3-CRL(K3)-ROW+1
     S(ROW,COL)=SMD(3,9)
4106 IF(RL(K4).NE.0)   GO TO 4107
     COL=K4-CRL(K4)-ROW+1
     S(ROW,COL)=SMD(3,10)
4107 IF(RL(K5).NE.0)   GO TO 4108
     COL=K5-CRL(K5)-ROW+1
     S(ROW,COL)=SMD(3,11)
4108 IF(RL(K6).NE.0)   GO TO 4109
     COL=K6-CRL(K6)-ROW+1
     S(ROW,COL)=SMD(3,12)
4109 IF(COL.GT.UBW) UBW=COL
4150 IF(RL(J4).NE.0)   GO TO  4200
     ROW=J4-CRL(J4)
     S(ROW,1)=S(ROW,1)+SMD(4,4)
     IF(RL(J5).NE.0)   GO TO 4151
     S(ROW,2)=S(ROW,2)+SMD(4,5)
4151 IF(RL(J6).NE.0)   GO TO 4152
     COL=J6-CRL(J6)-ROW+1
     S(ROW,COL)=S(ROW,COL)+SMD(4,6)
4152 IF(RL(K1).NE.0)   GO TO 4153
     COL=K1-CRL(K1)-ROW+1
     S(ROW,COL)=SMD(4,7)
4153 IF(RL(K2).NE.0)   GO TO 4154
     COL=K2-CRL(K2)-ROW+1
     S(ROW,COL)=SMD(4,8)
4154 IF(RL(K3).NE.0)   GO TO 4155
     COL=K3-CRL(K3)-ROW+1
     S(ROW,COL)=SMD(4,9)
4155 IF(RL(K4).NE.0)   GO TO 4156
```

```
      COL=K4-CRL(K4)-ROW+1
      S(ROW,COL)=SMD(4,10)
 4156 IF(RL(K5).NE.0)  GO TO 4157
      COL=K5-CRL(K5)-ROW+1
      S(ROW,COL)=SMD(4,11)
 4157 IF(RL(K6).NE.0)  GO TO 4158
      COL=K6-CRL(K6)-ROW+1
      S(ROW,COL)=SMD(4,12)
 4158 IF(COL.GT.UBW) UBW=COL
 4200 IF(RL(J5).NE.0)  GO TO 4250
      ROW=J5-CRL(J5)
      S(ROW,1)=S(ROW,1)+SMD(5,5)
      IF(RL(J6).NE.0)  GO TO 4201
      S(ROW,2)=S(ROW,2)+SMD(5,6)
 4201 IF(RL(K1).NE.0)  GO TO 4202
      COL=K1-CRL(K1)-ROW+1
      S(ROW,COL)=SMD(5,7)
 4202 IF(RL(K2).NE.0)  GO TO 4203
      COL=K2-CRL(K2)-ROW+1
      S(ROW,COL)=SMD(5,8)
 4203 IF(RL(K3).NE.0)  GO TO 4204
      COL=K3-CRL(K3)-ROW+1
      S(ROW,COL)=SMD(5,9)
 4204 IF(RL(K4).NE.0)  GO TO 4205
      COL=K4-CRL(K4)-ROW+1
      S(ROW,COL)=SMD(5,10)
 4205 IF(RL(K5).NE.0)  GO TO 4206
      COL=K5-CRL(K5)-ROW+1
      S(ROW,COL)=SMD(5,11)
 4206 IF(RL(K6).NE.0)  GO TO 4207
      COL=K6-CRL(K6)-ROW+1
      S(ROW,COL)=SMD(5,12)
 4207 IF(COL.GT.UBW) UBW=COL
 4250 IF(RL(J6).NE.0)  GO TO 4300
      ROW=J6-CRL(J6)
      S(ROW,1)=S(ROW,1)+SMD(6,6)
      IF(RL(K1).NE.0)  GO TO 4251
      COL=K1-CRL(K1)-ROW+1
      S(ROW,COL)=SMD(6,7)
 4251 IF(RL(K2).NE.0)  GO TO 4252
      COL=K2-CRL(K2)-ROW+1
      S(ROW,COL)=SMD(6,8)
 4252 IF(RL(K3).NE.0)  GO TO 4253
      COL=K3-CRL(K3)-ROW+1
      S(ROW,COL)=SMD(6,9)
 4253 IF(RL(K4).NE.0)  GO TO 4254
      COL=K4-CRL(K4)-ROW+1
      S(ROW,COL)=SMD(6,10)
```

```
4254 IF(RL(K5).NE.0)  GO TO 4255
     COL=K5-CRL(K5)-ROW+1
     S(ROW,COL)=SMD(6,11)
4255 IF(RL(K6).NE.0)  GO TO 4256
     COL=K6-CRL(K6)-ROW+1
     S(ROW,COL)=SMD(6,12)
4256 IF(COL.GT.UBW) UBW=COL
4300 IF(RL(K1).NE.0)  GO TO 4350
     ROW=K1-CRL(K1)
     S(ROW,1)=S(ROW,1)+SMD(7,7)
     IF(RL(K2).NE.0)  GO TO 4301
     S(ROW,2)=S(ROW,2)+SMD(7,8)
4301 IF(RL(K3).NE.0)  GO TO 4302
     COL=K3-CRL(K3)-ROW+1
     S(ROW,COL)=S(ROW,COL)+SMD(7,9)
4302 IF(RL(K4).NE.0)  GO TO 4303
     COL=K4-CRL(K4)-ROW+1
     S(ROW,COL)=S(ROW,COL)+SMD(7,10)
4303 IF(RL(K5).NE.0)  GO TO 4304
     COL=K5-CRL(K5)-ROW+1
     S(ROW,COL)=S(ROW,COL)+SMD(7,11)
4304 IF(RL(K6).NE.0)  GO TO 4350
     COL=K6-CRL(K6)-ROW+1
     S(ROW,COL)=S(ROW,COL)+SMD(7,12)
4350 IF(RL(K2).NE.0)  GO TO 4400
     ROW=K2-CRL(K2)
     S(ROW,1)=S(ROW,1)+SMD(8,8)
     IF(RL(K3).NE.0)  GO TO 4351
     S(ROW,2)=S(ROW,2)+SMD(8,9)
4351 IF(RL(K4).NE.0)  GO TO 4352
     COL=K4-CRL(K4)-ROW+1
     S(ROW,COL)=S(ROW,COL)+SMD(8,10)
4352 IF(RL(K5).NE.0)  GO TO 4353
     COL=K5-CRL(K5)-ROW+1
     S(ROW,COL)=S(ROW,COL)+SMD(8,11)
4353 IF(RL(K6).NE.0)  GO TO 4400
     COL=K6-CRL(K6)-ROW+1
     S(ROW,COL)=S(ROW,COL)+SMD(8,12)
4400 IF(RL(K3).NE.0)  GO TO 41
     ROW=K3-CRL(K3)
     S(ROW,1)=S(ROW,1)+SMD(9,9)
     IF(RL(K4).NE.0)  GO TO 4401
     S(ROW,2)=S(ROW,2)+SMD(9,10)
4401 IF(RL(K5).NE.0)  GO TO 4402
     COL=K5-CRL(K5)-ROW+1
     S(ROW,COL)=S(ROW,COL)+SMD(9,11)
4402 IF(RL(K6).NE.0)  GO TO 41
     COL=K6-CRL(K6)-ROW+1
```

```
      S(ROW,COL)=S(ROW,COL)+SMD(9,12)
   41 IF(RL(K4).NE.0)   GO TO 42
      ROW=K4-CRL(K4)
      S(ROW,1)=S(ROW,1)+SMD(10,10)
      IF(RL(K5).NE.0)   GO TO 411
      S(ROW,2)=S(ROW,2)+SMD(10,11)
  411 IF(RL(K6).NE.0)   GO TO 42
      COL=K6-CRL(K6)-ROW+1
      S(ROW,COL)=S(ROW,COL)+SMD(10,12)
   42 IF(RL(K5).NE.0)   GO TO 43
      ROW=K5-CRL(K5)
      S(ROW,1)=S(ROW,1)+SMD(11,11)
      IF(RL(K6).NE.0)   GO TO 43
      S(ROW,2)=S(ROW,2)+SMD(11,12)
   43 IF(RL(K6).NE.0) GO TO 2
      ROW=K6-CRL(K6)
      S(ROW,1)=S(ROW,1)+SMD(12,12)
      GO TO 2
   87 DO 88  I=1,N
      DO 88  J=1,N
      IF(ABS(S(I,J)).GT..0001) GO TO 89
      GO TO 88
   89 IF(J.GT.UBW) UBW=J
   88 CONTINUE
      IF(IFLAG.EQ.0) GO TO 60
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                CCCCCCCC
C
C  WRITE OUT ROW IREK OF PACKED STIFFNESS MATRIX
C
      DO 1 IREK=1,N
      WRITE(11'IREK)(S(IREK,J),J=1,UBW)
    1 CONTINUE
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                CCCCCCCC
   60 RETURN
      END
```

```
*DECK DCBAND
      SUBROUTINE DCBAND(N,UBW,A,I1,*,*)
      INTEGER UBW,P,Q
      DIMENSION A(I1,I1)
      DO 100 I=1,N
      P=N-I+1
      IF(UBW.LT.P) P=UBW
      DO 100 J=1,P
      Q=UBW-J
      IF(I-1.LT.Q) Q=I-1
      SUM=A(I,J)
      IF(Q.LT.1) GO TO 102
      DO 110 K=1,Q
      SUM=SUM-A(I-K,1+K)*A(I-K,J+K)
  110 CONTINUE
  102 IF(J.NE.1) GO TO 101
      IF(SUM.GT.0.0) GO TO 103
      RETURN 1
  103 TEMP=1.0/SQRT(SUM)
      A(I,J)=TEMP
      GO TO 100
  101 A(I,J)=SUM*TEMP
  100 CONTINUE
      RETURN 2
      END
```

```
*DECK LDATA
      SUBROUTINE LDATA
      COMMON DUM5(972),DUM6(39),IDUM3(44),IDUM4(11),
     1IX16,IX17,IX18,IX19,IX20,IX21,IX22,IX23,IX24,IX25,IX26,IX27,
     2IDUM7(306),
     5COLID1(48),COLID2(48),PMID1(38),PMID2(38),BRID1(20),BRID2(20)
      COMMON ITAB(300),ISEC(300),IFIRST,M,E,DJ(600),
     1X(100),Y(100),Z(100),L(300),AX(300),IX(300),IY(300),IZ(300),
     2AR(600),A(600),AC(600),AE(600),S(600,600),R(300,9),
     3AML(300,12),SM(12,12),SMD(12,12),SMR(12,12),AM(12),AMD(12),G,
     4JJ(300),JK(300),LML(300),JE,KE,TS,NDJ,NR,
     5RL(600),CRL(600),N,UBW,NJ,IX11,IX12,IX13,IX14,IX15,NROW,NUBW,
     6ST(360,360),D(600),
     1NLS,LN,IFLAG,IERR,ITRANS,ISTED
      INTEGER TS,RL,CRL,UBW,SN,AA
      REAL IX,IY,IZ,L
C
C           INPUT AND PRINT LOAD DATA
C           LOAD PARAMETERS
C
 400  LN=LN+1
C
C  ZERO OUT LOAD VECTORS
C
      DO 402 I=1,600
      A(I)=0.0
      AE(I)=0.0
      AC(I)=0.0
 402  CONTINUE
      DO 417 I=1,300
      LML(I)=0
      DO 401 J=1,12
      AML(I,J)=0.0
 401  CONTINUE
 417  CONTINUE
      WRITE(6,403) LN
  403 FORMAT(1X,/,1X,"LOADING NO.",I5/)
      PRINT 405
 405  FORMAT(" NLJ  NLM")
      READ(5,406)NLJ,NLM
 406  FORMAT(2I3)
      WRITE(6,407)NLJ,NLM
 407  FORMAT(" ",I3,I5)
      IF(NLJ.EQ.0)GO TO 425
C           ACTIONS APPLIED AT JOINTS
      PRINT 408
 408  FORMAT("0ACTIONS APPLIED AT JOINTS")
      PRINT 409
```

```
 409  FORMAT(" JOINT          A1          A2          ,A3          A4       LDA0030
     1    A5        A6")
      IF(TS.GE.3)GO TO 413                                                LDA0031
      DO 412 J=1,NLJ                                                      LDA0032
      READ(5,410)K,A(2*K-1),A(2*K)                                        LDA0033
 410  FORMAT(I3,2F10.3)                                                   LDA0034
      WRITE(6,411)K,A(2*K-1),A(2*K)                                       LDA0035
 411  FORMAT(" ",I4,F13.3,F12.3)                                          LDA0036
 412  CONTINUE                                                            LDA0037
      GO TO 425                                                           LDA0038
 413  IF(TS.EQ.6)GO TO 420                                                LDA0039
      DO 416 J=1,NLJ                                                      LDA0040
      READ(5,414)K,A(3*K-2),A(3*K-1),A(3*K)                               LDA0041
 414  FORMAT(I3,3F10.3)                                                   LDA0042
      WRITE(6,415)K,A(3*K-2),A(3*K-1),A(3*K)                              LDA0043
 415  FORMAT(" ",I4,F13.3,2F12.3)                                         LDA0044
 416  CONTINUE                                                            LDA0045
      GO TO 425                                                           LDA0046
 420  DO 424 J=1,NLJ                                                      LDA0047
      READ(5,421)K,A(6*K-5),A(6*K-4),A(6*K-3),A(6*K-2),A(6*K-1),A(6*K)    LDA0048
 421  FORMAT(I3,6F10.3)                                                   LDA0049
      WRITE(6,422)K,A(6*K-5),A(6*K-4),A(6*K-3),A(6*K-2),A(6*K-1),A(6*K)   LDA0050
 422  FORMAT(" ",I4,F13.3,5F12.3)                                         LDA0051
 424  CONTINUE                                                            LDA0052
C         ACTIONS AT ENDS OF RESTRAINED MEMBERS DUE TO LOADS
 425  IF(NLM.EQ.0)GO TO 500                                               LDA0053
      PRINT 426                                                           LDA0054
 426  FORMAT("0ACTIONS AT ENDS OF RESTRAINED MEMBERS DUE TO LOADS")       LDA0055
      PRINT 427                                                           LDA0056
 427  FORMAT(" MEMBER        AML1        AML2        AML3        AML4     LDA0057
     1   AML5        AML6")
      IF(TS.NE.6)GO TO 429                                                LDA0058
      PRINT 428                                                           LDA0059
 428  FORMAT("               AML7        AML8        AML9        AML10    LDA0060
     1   AML11       AML12")
 429  IF(TS.LT.3)GO TO 440                                                LDA0061
      DO 435 J=1,NLM                                                      LDA0062
      READ(5,430)I,AML(I,1),AML(I,2),AML(I,3),AML(I,4),AML(I,5),AML(I,6)LDA0063
 430  FORMAT(I3,6F10.3)                                                   LDA0064
      WRITE(6,431)I,AML(I,1),AML(I,2),AML(I,3),AML(I,4),AML(I,5),AML(I,6LDA0065
     1)
 431  FORMAT(" ",I4,F13.3,5F12.3)                                         LDA0066
      LML(I)=1                                                            LDA0067
      IF(TS.NE.6)GO TO 435                                                LDA0068
      READ(5,432)AML(I,7),AML(I,8),AML(I,9),AML(I,10),AML(I,11),          LDA0069
     1AML(I,12)
 432  FORMAT(6F10.3)                                                      LDA0070
      WRITE(6,433)AML(I,7),AML(I,8),AML(I,9),AML(I,10),AML(I,11),         LDA0071
```

```
     1AML(I,12)
 433  FORMAT(" ",F17.3,5F12.3)                                          LDA0072
 435  CONTINUE                                                          LDA0073
      GO TO 450                                                         LDA0074
 440  DO 445 J=1,NLM                                                    LDA0075
      READ(5,441)I,AML(I,1),AML(I,2),AML(I,3),AML(I,4)                  LDA0076
 441  FORMAT(I3,4F10.3)                                                 LDA0077
      WRITE(6,442)I,AML(I,1),AML(I,2),AML(I,3),AML(I,4)                 LDA0078
 442  FORMAT(" ",I4,F13.3,3F12.3)                                       LDA0079
      LML(I)=1                                                          LDA0080
 445  CONTINUE                                                          LDA0081
C          CONSTRUCTION OF VECTORS ASSOCIATED WITH LOADS
C                EQUIVALENT JOINT LOADS
 450  IF(TS.EQ.1)GO TO 460                                              LDA0082
      IF(TS.EQ.2)GO TO 470                                              LDA0083
      IF(TS.LT.6)GO TO 480                                              LDA0084
      GO TO 490                                                         LDA0085
 460  DO 465 I=1,M                                                      LDA0086
      IF(LML(I).NE.1)GO TO 465                                          LDA0087
      AE(2*I-1)=AE(2*I-1)-AML(I,1)                                      LDA0088
      AE(2*I)=AE(2*I)-AML(I,2)                                          LDA0089
      AE(2*I+1)=AE(2*I+1)-AML(I,3)                                      LDA0090
      AE(2*I+2)=AE(2*I+2)-AML(I,4)                                      LDA0091
 465  CONTINUE                                                          LDA0092
      GO TO 500                                                         LDA0093
 470  DO 475 I=1,M                                                      LDA0094
      IF(LML(I).NE.1)GO TO 475                                          LDA0095
      JJT=JJ(I)                                                         LDA0096
      JKT=JK(I)                                                         LDA0097
      AE(2*JJT   -1)=AE(2*JJT   -1)-R(I,1)*AML(I,1)-R(I,3)*AML(I,2)     LDA0098
      AE(2*JJT   )=AE(2*JJT   )-R(I,2)*AML(I,1)-R(I,4)*AML(I,2)         LDA0099
      AE(2*JKT   -1)=AE(2*JKT   -1)-R(I,1)*AML(I,3)-R(I,3)*AML(I,4)     LDA0100
      AE(2*JKT   )=AE(2*JKT   )-R(I,2)*AML(I,3)-R(I,4)*AML(I,4)         LDA0101
 475  CONTINUE                                                          LDA0102
      GO TO 500                                                         LDA0103
 480  DO 485 I=1,M                                                      LDA0104
      IF(LML(I).NE.1)GO TO 485                                          LDA0105
      JJT=JJ(I)                                                         LDA0106
      JKT=JK(I)                                                         LDA0107
      AE(3*JJT   -2)=AE(3*JJT   -2)-R(I,1)*AML(I,1)-R(I,4)*AML(I,2)-R(I,7)LDA0108
     1*AML(I,3)
      AE(3*JJT   -1)=AE(3*JJT   -1)-R(I,2)*AML(I,1)-R(I,5)*AML(I,2)-R(I,8)LDA0109
     1*AML(I,3)
      AE(3*JJT   )=AE(3*JJT   )-R(I,3)*AML(I,1)-R(I,6)*AML(I,2)-R(I,9)*AMLLDA0110
     1(I,3)
      AE(3*JKT   -2)=AE(3*JKT   -2)-R(I,1)*AML(I,4)-R(I,4)*AML(I,5)-R(I,7)LDA0111
     1*AML(I,6)
      AE(3*JKT   -1)=AE(3*JKT   -1)-R(I,2)*AML(I,4)-R(I,5)*AML(I,5)-R(I,8)LDA0112
```

```
     1*AML(I,6)
      AE(3*JKT   )=AE(3*JKT   )-R(I,3)*AML(I,4)-R(I,6)*AML(I,5)-R(I,9)*AMLLDA0113
     1(I,6)
  485 CONTINUE                                                          LDA0114
      GO TO 500                                                         LDA0115
  490 DO 495 I=1,M                                                      LDA0116
      IF(LML(I).NE.1)GO TO 495                                          LDA0117
      JJT=JJ(I)                                                         LDA0118
      JKT=JK(I)                                                         LDA0119
      AE(6*JJT   -5)=AE(6*JJT   -5)-R(I,1)*AML(I,1)-R(I,4)*AML(I,2)-R(I,7)LDA0120
     1*AML(I,3)
      AE(6*JJT   -4)=AE(6*JJT   -4)-R(I,2)*AML(I,1)-R(I,5)*AML(I,2)-R(I,8)LDA0121
     1*AML(I,3)
      AE(6*JJT   -3)=AE(6*JJT   -3)-R(I,3)*AML(I,1)-R(I,6)*AML(I,2)-R(I,9)LDA0122
     1*AML(I,3)
      AE(6*JJT   -2)=AE(6*JJT   -2)-R(I,1)*AML(I,4)-R(I,4)*AML(I,5)-R(I,7)LDA0123
     1*AML(I,6)
      AE(6*JJT   -1)=AE(6*JJT   -1)-R(I,2)*AML(I,4)-R(I,5)*AML(I,5)-R(I,8)LDA0124
     1*AML(I,6)
      AE(6*JJT   )=AE(6*JJT   )-R(I,3)*AML(I,4)-R(I,6)*AML(I,5)-R(I,9)*AMLLDA0125
     1(I,6)
      AE(6*JKT   -5)=AE(6*JKT   -5)-R(I,1)*AML(I,7)-R(I,4)*AML(I,8)-R(I,7)LDA0126
     1*AML(I,9)
      AE(6*JKT   -4)=AE(6*JKT   -4)-R(I,2)*AML(I,7)-R(I,5)*AML(I,8)-R(I,8)LDA0127
     1*AML(I,9)
      AE(6*JKT   -3)=AE(6*JKT   -3)-R(I,3)*AML(I,7)-R(I,6)*AML(I,8)-R(I,9)LDA0128
     1*AML(I,9)
      AE(6*JKT   -2)=AE(6*JKT   -2)-R(I,1)*AML(I,10)-R(I,4)*AML(I,11)-R(I,LDA0129
     17)*AML(I,12)
      AE(6*JKT   -1)=AE(6*JKT   -1)-R(I,2)*AML(I,10)-R(I,5)*AML(I,11)-R(I,LDA0130
     18)*AML(I,12)
      AE(6*JKT   )=AE(6*JKT   )-R(I,3)*AML(I,10)-R(I,6)*AML(I,11)-R(I,9)*ALDA0131
     1ML(I,12)
  495 CONTINUE                                                          LDA0132
C          COMBINED JOINT LOADS
  500 NPNR=N+NR                                                         LDA0133
      DO 505 J=1,NPNR                                                   LDA0134
      IF(RL(J).NE.0)GO TO 501                                           LDA0135
      K=J-CRL(J)                                                        LDA0136
      GO TO 502                                                         LDA0137
  501 K=N+CRL(J)                                                        LDA0138
  502 AC(K)=A(J)+AE(J)                                                  LDA0139
  505 CONTINUE                                                          LDA0140
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                               CCCCCCCC
C
C    WRITE OUT LOAD VECTOR LN TO DISK
C
```

```
      WRITE(15"LN)(AC(I),I=1,N)                                         LDA0141
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                               CCCCCCCC
      RETURN                                                            LDA0142
      END                                                               LDA0143
```

```
*DECK RESULT
      SUBROUTINE RESULT
      COMMON DUM5(972),DUM6(39),IDUM3(44),IDUM4(11),
     1IX16,IX17,IX18,IX19,IX20,IX21,IX22,IX23,IX24,IX25,IX26,IX27,
     2IDUM7(306),
     5COLID1(48),COLID2(48),BMID1(38),BMID2(38),BRID1(20),BRID2(20)
      COMMON ITAB(300),ISEC(300),IFIRST,M,E,DJ(600),
     1X(100),Y(100),Z(100),L(300),AX(300),IX(300),IY(300),IZ(300),
     2AR(600),A(600),AC(600),AE(600),S(600,600),R(300,9),
     3AML(300,12),SM(12,12),SMD(12,12),SMP(12,12),AM(12),AMD(12),G,
     4JJ(300),JK(300),LML(300),JE,KE,TS,NDJ,NR,
     5RL(600),CRL(600),N,UPW,NJ,IX11,IX12,IX13,IX14,IX15,NROW,NUBW,
     6ST(360,360),D(600),
     1NLS,LN,IFLAG,IEPR,ITRANS,ISTEP
      INTEGER TS,RL,CRL,UBW
      REAL IX,IY,IZ,L
C
C   ZERO OUT REACTION VECTOR
C
      DO 418 I=1,600
      AR(I)=0.0
  418 CONTINUE
      PRINT 99
   99 FORMAT("0JOINT DISPLACEMENTS")
      PRINT 3
  3   FORMAT(" JOINT            D1             D2             D3             D4
     1          D5             D6")
      J=N+1
      NPNR=N+NR
      DO 41 K=1,NPNR
      JE=NPNR-K+1
      IF(RL(JE).EQ.0)  GO TO 42
      DJ(JE)=0.0
      GO TO 41
   42 J=J-1
      DJ(JE)=D(J)
   41 CONTINUE
      IF(TS.GE.3)  GO TO 50
      NJ2=2*NJ
      DO 52 JE=2,NJ2,2
      JE2=JE/2
      WRITE(6,521)JE2,DJ(JE-1),DJ(JE)
  521 FORMAT(" ",I4,1PE16.5,1PE12.5)
   52 CONTINUE
      GO TO 1
   50 IF(TS.EQ.6)GO TO 51
      NJ3=3*NJ
      DO 511 JE=3,NJ3,3
```

```
      JE3=JE/3
      WRITE(6,512)JE3,DJ(JE-2),DJ(JE-1),DJ(JE)
512   FORMAT(" ",I4,1PE16.5,1P2E12.5)
511   CONTINUE
      GO TO 1
51    NJ6=6*NJ
      DO 513 JE=6,NJ6,6
      JE6=JE/6
      WRITE(6,514)JE6,DJ(JE-5),DJ(JE-4),DJ(JE-3),DJ(JE-2),DJ(JE-1),DJ(JE
     1)
514   FORMAT(" ",I4,1PE16.5,1P5E12.5)
513   CONTINUE
1     PRINT 11
11    FORMAT("0MEMBER END ACTIONS")
      PRINT 12
12    FORMAT(" MEMBER            AM1             AM2             AM3             AM4
     1    AM5             AM6")
      IF(TS.EQ.6)GO TO 13
      GO TO 15
13    PRINT 14
14    FORMAT("                   AM7             AM8             AM9             AM10
     1    AM11            AM12")
   15 I=0
    4 I=I+1
      IF(I.GT.M)  GO TO 100
      IF(TS.GE.3)  GO TO 61
      J1=2*JJ(I)-1
      J2=2*JJ(I)
      K1=2*JK(I)-1
      K2=2*JK(I)
      GO TO 63
   61 IF(TS.EQ.6)  GO TO 62
      J1=3*JJ(I)-2
      J2=3*JJ(I)-1
      J3=3*JJ(I)
      K1=3*JK(I)-2
      K2=3*JK(I)-1
      K3=3*JK(I)
      GO TO 63
   62 J1=6*JJ(I)-5
      J2=6*JJ(I)-4
      J3=6*JJ(I)-3
      J4=6*JJ(I)-2
      J5=6*JJ(I)-1
      J6=6*JJ(I)
      K1=6*JK(I)-5
      K2=6*JK(I)-4
      K3=6*JK(I)-3
```

```
      K4=6*JK(I)-2
      K5=6*JK(I)-1
      K6=6*JK(I)
   63 IF(TS.EQ.1.OR.TS.EQ.3.OR.TS.EQ.6) GO TO 631
   64 IF(TS.EQ.2.OR.TS.EQ.3.OR.TS.EQ.5.OR.TS.EQ.6) GO TO 641
   65 IF(TS.EQ.4.OR.TS.EQ.6)  GO TO 651
      GO TO 7
  631 SCM2Z=(4.0*E*IZ(I))/L(I)
      SCM3Z=(1.5*SCM2Z)/L(I)
      SCM4Z=(2.0*SCM3Z)/L(I)
      GO TO 64
  641 SCM1A=(E*AX(I))/L(I)
      GO TO 65
  651 SCM1B=(G*IX(I))/L(I)
      SCM2Y=(4.0*E*IY(I))/L(I)
      SCM3Y=(1.5*SCM2Y)/L(I)
      SCM4Y=(2.0*SCM3Y)/L(I)
    7 IF(TS.EQ.1)   GO  TO 80
      IF(TS.EQ.2)   GO  TO 81
      IF(TS.EQ.3)   GO  TO 82
      IF(TS.EQ.4)   GO  TO 83
      IF(TS.EQ.5)   GO  TO 84
      IF(TS.EQ.6)   GO  TO 85
   80 SMR(1,1)=SCM4Z
      SMR(3,3)=SCM4Z
      SMR(1,3)=-SCM4Z
      SMR(3,1)=-SCM4Z
      SMR(1,2)=SCM3Z
      SMR(2,1)=SCM3Z
      SMR(1,4)=SCM3Z
      SMR(4,1)=SCM3Z
      SMR(2,3)=-SCM3Z
      SMR(3,2)=-SCM3Z
      SMR(3,4)=-SCM3Z
      SMR(4,3)=-SCM3Z
      SMR(2,2)=SCM2Z
      SMR(4,4)=SCM2Z
      SMR(2,4)=SCM2Z/2.0
      SMR(4,2)=SCM2Z/2.0
      GO TO 10
   81 SM(1,1)=SCM1A
      SM(3,3)=SCM1A
      SM(1,3)=-SCM1A
      SM(3,1)=-SCM1A
      GO TO 9
   82 SM(1,1)=SCM1A
      SM(4,4)=SCM1A
      SM(1,4)=-SCM1A
```

```
      SM(4,1)=-SCM1A
      SM(2,2)=SCM4Z
      SM(5,5)=SCM4Z
      SM(2,5)=-SCM4Z
      SM(5,2)=-SCM4Z
      SM(2,3)=SCM3Z
      SM(3,2)=SCM3Z
      SM(2,6)=SCM3Z
      SM(6,2)=SCM3Z
      SM(3,5)=-SCM3Z
      SM(5,3)=-SCM3Z
      SM(5,6)=-SCM3Z
      SM(6,5)=-SCM3Z
      SM(3,3)=SCM2Z
      SM(6,6)=SCM2Z
      SM(3,6)=SCM2Z/2.0
      SM(6,3)=SCM2Z/2.0
      GO TO 19
   83 SM(1,1)=SCM1B
      SM(4,4)=SCM1B
      SM(1,4)=-SCM1B
      SM(4,1)=-SCM1B
      SM(2,2)=SCM2Y
      SM(5,5)=SCM2Y
      SM(2,5)=SCM2Y/2.0
      SM(5,2)=SCM2Y/2.0
      SM(2,6)=SCM3Y
      SM(6,2)=SCM3Y
      SM(5,6)=SCM3Y
      SM(6,5)=SCM3Y
      SM(2,3)=-SCM3Y
      SM(3,2)=-SCM3Y
      SM(3,5)=-SCM3Y
      SM(5,3)=-SCM3Y
      SM(3,3)=SCM4Y
      SM(6,6)=SCM4Y
      SM(3,6)=-SCM4Y
      SM(6,3)=-SCM4Y
      GO TO 19
   84 SM(1,1)=SCM1A
      SM(4,4)=SCM1A
      SM(1,4)=-SCM1A
      SM(4,1)=-SCM1A
      GO TO 19
   85 SM(1,1)=SCM1A
      SM(7,7)=SCM1A
      SM(1,7)=-SCM1A
      SM(7,1)=-SCM1A
```

```
      SM(2,2)=SCM4Z
      SM(8,8)=SCM4Z
      SM(2,8)=-SCM4Z
      SM(8,2)=-SCM4Z
      SM(2,6)=SCM3Z
      SM(6,2)=SCM3Z
      SM(2,12)=SCM3Z
      SM(12,2)=SCM3Z
      SM(6,8)=-SCM3Z
      SM(8,6)=-SCM3Z
      SM(8,12)=-SCM3Z
      SM(12,8)=-SCM3Z
      SM(3,3)=SCM4Y
      SM(9,9)=SCM4Y
      SM(3,9)=-SCM4Y
      SM(9,3)=-SCM4Y
      SM(3,5)=-SCM3Y
      SM(5,3)=-SCM3Y
      SM(3,11)=-SCM3Y
      SM(11,3)=-SCM3Y
      SM(4,4)=SCM1B
      SM(10,10)=SCM1B
      SM(4,10)=-SCM1B
      SM(10,4)=-SCM1B
      SM(5,5)=SCM2Y
      SM(11,11)=SCM2Y
      SM(5,11)=SCM2Y/2.0
      SM(11,5)=SCM2Y/2.0
      SM(5,9)=SCM3Y
      SM(9,5)=SCM3Y
      SM(9,11)=SCM3Y
      SM(11,9)=SCM3Y
      SM(6,6)=SCM2Z
      SM(12,12)=SCM2Z
      SM(6,12)=SCM2Z/2.0
      SM(12,6)=SCM2Z/2.0
      GO TO 19
    9 DO 91 K=1,2
      DO 91 J=1,4
      SMR(J,2*K-1)=SM(J,2*K-1)*R(I,1)+SM(J,2*K)*R(I,3)
      SMR(J,2*K)=SM(J,2*K-1)*R(I,2)+SM(J,2*K)*R(I,4)
   91 CONTINUE
   10 DO 101 J=1,4
      AMD(J)=SMR(J,1)*DJ(J1)+SMR(J,2)*DJ(J2)+SMR(J,3)*DJ(K1)
     1+SMR(J,4)*DJ(K2)
  101 CONTINUE
      DO 111 J=1,4
      AM(J)=AML(I,J)+AMD(J)
```

```
  111 CONTINUE
      WRITE (6,102) I,AM(1),AM(2),AM(3),AM(4)
 102  FORMAT(" ",I4,1PE16.5,1P3E12.5)
      IF(TS.NE.1)  GO TO 17
      IF(RL(J1).NE.1)  GO TO 141
      AR(J1)=AR(J1)+AMD(1)
  141 IF(RL(J2).NE.1)  GO TO 151
      AR(J2)=AR(J2)+AMD(2)
  151 IF(RL(K1).NE.1)  GO TO 16
      AR(K1)=AR(K1)+AMD(3)
   16 IF(RL(K2).NE.1)  GO TO 4
      AR(K2)=AR(K2)+AMD(4)
      GO TO 4
   17 IF(RL(J1).NE.1)  GO TO 171
      AR(J1)=AR(J1)+R(I,1)*AMD(1)+R(I,3)*AMD(2)
  171 IF(RL(J2).NE.1)  GO TO 172
      AR(J2)=AR(J2)+R(I,2)*AMD(1)+R(I,4)*AMD(2)
  172 IF(RL(K1).NE.1)  GO TO 173
      AR(K1)=AR(K1)+R(I,1)*AMD(3)+R(I,3)*AMD(4)
  173 IF(RL(K2).NE.1)  GO TO 4
      AR(K2)=AR(K2)+R(I,2)*AMD(3)+R(I,4)*AMD(4)
      GO TO 4
   19 KL=2*NDJ/3
      JL=2*NDJ
      DO 191 K=1,KL
      DO 191 J=1,JL
      SMR(J,3*K-2)=SM(J,3*K-2)*R(I,1)+SM(J,3*K-1)*R(I,4)+SM(J,3*K)*R(I
     1,7)
      SMR(J,3*K-1)=SM(J,3*K-2)*R(I,2)+SM(J,3*K-1)*R(I,5)+SM(J,3*K)*R(I
     1,8)
      SMR(J,3*K)=SM(J,3*K-2)*R(I,3)+SM(J,3*K-1)*R(I,6)+SM(J,3*K)*R(I,9)
  191 CONTINUE
      IF(TS.EQ.6)  GO TO  24
      DO 21  J=1,6
   21 AMD(J)=SMR(J,1)*DJ(J1)+SMR(J,2)*DJ(J2)+SMR(J,3)*DJ(J3)+SMR(J,4)*
     1DJ(K1)+SMR(J,5)*DJ(K2)+SMR(J,6)*DJ(K3)
      DO 22  J=1,6
      AM(J)=AML(I,J)+AMD(J)
   22 CONTINUE
      WRITE  (6,221)  I,AM(1),AM(2),AM(3),AM(4),AM(5),AM(6)
 221  FORMAT(" ",I4,1PE16.5,1P5E12.5)
      IF(RL(J1).NE.1)  GO TO 231
      AR(J1)=AR(J1)+R(I,1)*AMD(1)+R(I,4)*AMD(2)+R(I,7)*AMD(3)
  231 IF(RL(J2).NE.1)  GO TO 232
      AR(J2)=AR(J2)+R(I,2)*AMD(1)+R(I,5)*AMD(2)+R(I,8)*AMD(3)
  232 IF(RL(J3).NE.1)  GO TO 233
      AR(J3)=AR(J3)+R(I,3)*AMD(1)+R(I,6)*AMD(2)+R(I,9)*AMD(3)
  233 IF(RL(K1).NE.1)  GO TO 234
```

```
      AR(K1)=AR(K1)+R(I,1)*AMD(4)+R(I,4)*AMD(5)+R(I,7)*AMD(6)           RES0267
  234 IF(RL(K2).NE.1)  GO TO 235                                        RES0268
      AR(K2)=AR(K2)+R(I,2)*AMD(4)+R(I,5)*AMD(5)+R(I,8)*AMD(6)           RES0269
  235 IF(RL(K3).NE.1)  GO TO 4                                          RES0270
      AR(K3)=AR(K3)+R(I,3)*AMD(4)+R(I,6)*AMD(5)+R(I,9)*AMD(6)           RES0271
      GO TO 4                                                           RES0272
   24 DO 241 J=1,12                                                     RES0273
      AMD(J)=SMR(J,1)*DJ(J1)+SMR(J,2)*DJ(J2)+SMR(J,3)*DJ(J3)+SMR(J,4)*  RES0274
     1DJ(J4)+SMR(J,5)*DJ(J5)+SMR(J,6)*DJ(J6)+SMR(J,7)*DJ(K1)+SMR(J,8)*
     1DJ(K2)+SMR(J,9)*DJ(K3)+SMR(J,10)*DJ(K4)+SMR(J,11)*DJ(K5)
     1+SMR(J,12)*DJ(K6)
  241 CONTINUE                                                          RES0275
      DO 242 J=1,12                                                     RES0276
      AM(J)=AML(I,J)+AMD(J)                                             RES0277
  242 CONTINUE                                                          RES0278
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                               CCCCCCCC
C
C  APPROPRIATE MEMBER END ACTIONS ARE WRITTEN OUT TO DISK
C
      JFILE=20+LN                                                       RES0279
      WRITE(JFILE'I) AM(1),AM(5),AM(6),AM(11),AM(12)                    RES0280
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                               CCCCCCCC
      WRITE (6,243) I,AM(1),AM(2),AM(3),AM(4),AM(5),AM(6)               RES0281
      WRITE (6,244) AM(7),AM(8),AM(9),AM(10),AM(11),AM(12)              RES0282
 243  FORMAT(" ",I4,1PE16.5,1P5E12.5)                                   RES0283
 244  FORMAT(" ",1PE20.5,1P5E12.5)                                      RES0284
         IF(RL(J1).NE.1)  GO TO 252                                     RES0285
      AR(J1)=AR(J1)+R(I,1)*AMD(1)+R(I,4)*AMD(2)+R(I,7)*AMD(3)           RES0286
  252    IF(RL(J2).NE.1)  GO TO 253                                     RES0287
      AR(J2)=AR(J2)+R(I,2)*AMD(1)+R(I,5)*AMD(2)+R(I,8)*AMD(3)           RES0288
  253    IF(RL(J3).NE.1)  GO TO 254                                     RES0289
      AR(J3)=AR(J3)+R(I,3)*AMD(1)+R(I,6)*AMD(2)+R(I,9)*AMD(3)           RES0290
  254    IF(RL(J4).NE.1)  GO TO 255                                     RES0291
      AR(J4)=AR(J4)+R(I,1)*AMD(4)+R(I,4)*AMD(5)+R(I,7)*AMD(6)           RES0292
  255    IF(RL(J5).NE.1)  GO TO 256                                     RES0293
      AR(J5)=AR(J5)+R(I,2)*AMD(4)+R(I,5)*AMD(5)+R(I,8)*AMD(6)           RES0294
  256    IF(RL(J6).NE.1)  GO TO 257                                     RES0295
      AR(J6)=AR(J6)+R(I,3)*AMD(4)+R(I,6)*AMD(5)+R(I,9)*AMD(6)           RES0296
  257    IF(RL(K1).NE.1)  GO TO 258                                     RES0297
      AR(K1)=AR(K1)+R(I,1)*AMD(7)+R(I,4)*AMD(8)+R(I,7)*AMD(9)           RES0298
  258    IF(RL(K2).NE.1)  GO TO 259                                     RES0299
      AR(K2)=AR(K2)+R(I,2)*AMD(7)+R(I,5)*AMD(8)+R(I,8)*AMD(9)           RES0300
  259    IF(RL(K3).NE.1)  GO TO 260                                     RES0301
      AR(K3)=AR(K3)+R(I,3)*AMD(7)+R(I,6)*AMD(8)+R(I,9)*AMD(9)           RES0302
  260    IF(RL(K4).NE.1)  GO TO 261                                     RES0303
```

```
      AR(K4)=AR(K4)+R(I,1)*AMD(10)+R(I,4)*AMD(11)+R(I,7)*AMD(12)          RES0304
  261    IF(RL(K5).NE.1)  GO TO 262                                       RES0305
      AR(K5)=AR(K5)+R(I,2)*AMD(10)+R(I,5)*AMD(11)+R(I,8)*AMD(12)          RES0306
  262    IF(RL(K6).NE.1)  GO TO 4                                         RES0307
      AR(K6)=AR(K6)+R(I,3)*AMD(10)+R(I,6)*AMD(11)+R(I,9)*AMD(12)          RES0308
      GO TO 4                                                             RES0309
  100 WRITE (6,540)                                                       RES0310
 540  FORMAT("0SUPPORT REACTIONS")                                        RES0311
      WRITE (6,541)                                                       RES0312
  541 FORMAT (1X,"JOINT",10X,"AR1",10X,"AR2",10X,"AR3",10X,"AR4",10X,     RES0313
     1"AR5",10X,"AR6")
      NPNR=N+NP                                                           RES0314
      DO 20 K=1,NPNR                                                      RES0315
      IF(RL(K).EQ.1) AR(K)=AR(K)-A(K)-AE(K)                               RES0316
   20 CONTINUE                                                            RES0317
      IF(TS.LT.3)  GO TO 551                                              RES0318
      IF(TS.EQ.6) GO TO 560                                               RES0319
      NJ3=3*NJ                                                            RES0320
      DO 555 KE=3,NJ3,3                                                   RES0321
      IF(RL(KE-2).EQ.1)  GO TO 556                                        RES0322
      IF(RL(KE-1).EQ.1)  GO TO 556                                        RES0323
      IF(RL(KE).EQ.1)  GO TO 556                                          RES0324
      GO TO 555                                                           RES0325
 556  KE3=KE/3                                                            RES0326
      WRITE (6,557)KE3,AR(KE-2),AR(KE-1),AR(KE)                           RES0327
 557  FORMAT(" ",I4,1PE16.5,1P2E12.5)                                     RES0328
  555 CONTINUE                                                            RES0329
      GO TO 600                                                           RES0330
  551 NJ2=2*NJ                                                            RES0331
      DO 552 KE=2,NJ2,2                                                   RES0332
      IF(RL(KE-1).EQ.1.OR.RL(KE).EQ.1)  GO TO 553                         RES0333
      GO TO 552                                                           RES0334
  553 KE2=KE/2                                                            RES0335
      WRITE (6,554)KE2,AR(KE-1),AR(KE)                                    RES0336
 554  FORMAT(" ",I4,1PE16.5,1PE12.5)                                      RES0337
  552 CONTINUE                                                            RES0338
      GO TO 600                                                           RES0339
  560 NJ6=6*NJ                                                            RES0340
      DO 561 KE=6,NJ6,6                                                   RES0341
      IF(RL(KE-5).EQ.1)  GO TO 562                                        RES0342
      IF(RL(KE-4).EQ.1)  GO TO 562                                        RES0343
      IF(RL(KE-3).EQ.1)  GO TO 562                                        RES0344
      IF(RL(KE-2).EQ.1)  GO TO 562                                        RES0345
      IF(RL(KE-1).EQ.1)  GO TO 562                                        RES0346
      IF(RL(KE-0).EQ.1)  GO TO 562                                        RES0347
      GO TO 561                                                           RES0348
  562 KE6=KE/6                                                            RES0349
      WRITE (6,563)KE6,AR(KE-5),AR(KE-4),AR(KE-3),AR(KE-2),AR(KE-1),      RES0350
```

```
     1AR(KE)                                                    
563  FORMAT(" ",I4,1PE16.5,1P5E12.5)                            RES0351
 561 CONTINUE                                                   RES0352
 600 RETURN                                                     RES0353
     END                                                        RES0354
```

```
*DECK SBAND
      SUBROUTINE SBAND(N,UBW,U,B,X,J1)
      DIMENSION U(J1,J1),B(600),X(600)
      INTEGER UBW
      DO 99 I=1,600
      X(I)=0.0
   99 CONTINUE
      DO 100 I=1,N
      J=I-UBW+1
      IF (I+1.LE.UBW) J=1
      SUM=B(I)
      I1=I-1
      IF (J.GT.I1) GO TO 101
      DO 110 K=J,I1
      SUM=SUM-U(K,I-K+1)*X(K)
  110 CONTINUE
  101 X(I)=SUM*U(I,1)
  100 CONTINUE
      DO 150 I1=1,N
      I=N-I1+1
      J=I+UBW-1
      IF (J.GT.N) J=N
      SUM=X(I)
      K1=I+1
      IF (K1.GT.J) GO TO 151
      DO 160 K=K1,J
      SUM=SUM-U(I,K-I+1)*X(K)
  160 CONTINUE
  151 X(I)=SUM*U(I,1)
  150 CONTINUE
      RETURN
      END
```

APPENDIX G

LISTINGS OF SAMPLE INPUT

This Appendix contains listings of input data for Examples A2, BW2, and the Kinematic Condensation Example, all of which are discussed in detail in Chapter 5. The input formats for this data are detailed in Appendix A.

Each example input listing begins with a line starting with an identifier *DECK followed by the name of the example. This line is not part of the input.

```
*DECK 24 STORY 3 BAY UNBRACED FRAME EXAMPLE A2
PLANE FRAME --- 24 STORY 3 BAY UNBRACED
   48    38    20     0    10     2   100     1     0     0     5     3   100.00   0.00  10.00
    3     2     1     2
   97     0
      1.00        0.00        0.00        0.00
      6.91     1
    4     2     1     2
  100     0
      1.00        0.00        0.00        0.00
      6.91     1
    5     2     1     2
   69     0
      1.00        0.00        0.00        0.00
      4.90     1
   6WF20       1       5.90       0.24      13.30      41.70
   8WF24       2       7.06       0.34      18.20      82.50
   8WF28       3       8.23       0.53      21.60      97.80
   8WF31       4       9.12       0.53      37.00     109.70
   8WF35       5      10.30       0.77      42.50     126.50
  10WF39       6      11.48       0.37      44.90     209.70
  12WF40       7      11.77       0.96      44.10     310.10
  14WF43       8      12.65       1.05      45.10     429.10
  14WF48       9      14.11       1.44      51.30     464.90
  14WF53      10      15.59       1.93      57.50     542.10
  12WF58      11      17.06       2.10     107.40     476.10
  14WF61      12      17.94       2.19     107.30     641.50
  14WF74      13      21.76       3.86     133.50     796.80
  14WF78      14      22.94       3.52     206.90     851.20
  12WF79      15      23.22       3.85     216.40     663.00
  14WF84      16      24.71       4.41     225.50     928.40
  12WF99      17      29.09       7.45     278.20     858.50
 14WF111      18      32.65       7.48     454.90    1266.50
 14WF119      19      34.99       9.20     491.80    1373.10
```

14WF127	20	37.33	11.10	527.60	1476.70
14WF136	21	39.98	13.50	567.70	1593.00
14WF142	22	41.85	14.20	660.10	1672.20
14WF150	23	44.08	16.70	702.50	1786.90
14WF158	24	46.47	19.50	745.00	1900.60
14WF167	25	49.09	22.80	790.20	2020.80
14WF176	26	51.73	26.50	837.90	2149.60
14WF184	27	54.07	30.30	882.70	2274.80
14WF193	28	56.73	34.70	930.10	2402.40
14WF202	29	59.39	39.60	979.70	2538.80
14WF211	30	62.07	44.80	1028.60	2671.40
14WF219	31	64.36	49.90	1073.20	2798.20
14WF228	32	67.06	56.20	1124.80	2942.40
14WF237	33	69.69	62.60	1174.80	3080.90
14WF246	34	72.33	69.70	1226.60	3228.90
14WF264	35	77.63	85.30	1331.20	3526.00
14WF287	36	84.37	108.00	1466.50	3912.10
14WF314	37	92.30	140.00	1631.40	4399.40
14WF320	38	94.12	137.00	1635.10	4141.70
14WF342	39	100.59	178.00	1806.90	4911.50
14WF370	40	108.78	222.00	1986.00	5454.20
14WF398	41	116.98	272.00	2169.70	6013.70
14WF426	42	125.25	330.00	2359.50	6610.30
14WF455	43	133.73	396.00	2561.20	7214.90
14WF500	44	146.95	514.00	2882.70	8234.10
14WF550	45	161.75	670.00	3256.70	9443.10
14WF605	46	177.85	869.00	3680.90	10842.30
14WF665	47	195.51	1120.00	4166.20	12477.70
14WF730	48	214.65	1450.00	4716.80	14371.40
6JR4.4	1	1.30	0.01	0.17	7.30
8JR6.5	2	1.92	0.02	0.34	18.70
10JR9	3	2.64	0.03	0.61	39.00
12JR11.8	4	3.45	0.04	0.98	72.00
10B15	5	4.40	0.10	2.79	68.80
12B16.5	6	4.86	0.11	2.79	105.30

14B17.2	7	5.05	0.11	2.65	147.30
14B22	8	6.47	0.21	6.40	197.40
16B26	9	7.65	0.26	8.71	298.10
14WF30	10	8.81	0.38	17.50	289.60
16B31	11	9.12	0.46	11.57	372.50
14WF34	12	10.00	0.57	21.30	339.20
16WF36	13	10.59	0.55	22.10	446.30
16WF40	14	11.77	0.79	26.50	515.50
18WF45	15	13.24	0.89	31.90	704.50
18WF50	16	14.71	1.25	37.20	800.60
21WF55	17	16.18	1.24	44.00	1140.70
21WF62	18	18.23	1.83	53.10	1326.80
24WF68	19	20.00	1.86	63.80	1814.50
24WF76	20	22.37	2.70	76.50	2096.40
27WF84	21	24.71	2.79	95.70	2624.80
27WF94	22	27.65	4.06	115.10	3266.70
30WF99	23	29.11	3.78	116.90	3988.60
30WF108	24	31.77	5.02	135.10	4461.00
30WF116	25	34.13	6.43	153.20	4919.10
33WF118	26	34.71	5.32	170.30	5886.90
33WF130	27	38.26	7.37	201.40	6699.00
36WF135	28	39.70	7.03	207.10	7796.10
36WF150	29	44.16	10.10	250.40	9012.10
36WF160	30	47.09	12.40	275.40	9738.80
36WF170	31	49.98	15.10	300.60	10470.00
36WF182	32	53.54	18.40	327.70	11281.50
36WF194	33	57.11	22.30	355.40	12103.40
36WF230	34	67.73	28.60	870.90	14988.40
36WF245	35	72.03	34.70	944.70	16092.20
36WF260	36	76.56	41.60	1020.60	17233.80
36WF280	37	82.32	52.60	1127.50	18819.30
36WF300	38	88.17	64.20	1225.20	20290.20
3UAN6.1	1	1.80	0.01	0.01	0.01
3UAN9.0	2	2.62	0.01	0.01	0.01
4UAN11.6	3	3.38	0.01	0.01	0.01

3UAN13.2	4	3.84	0.01	0.01	0.01
4UAN14.4	5	4.18	0.01	0.01	0.01
4UAN15.4	6	4.50	0.01	0.01	0.01
4UAN17.0	7	4.96	0.01	0.01	0.01
4UAN18.2	8	5.34	0.01	0.01	0.01
4UAN19.6	9	5.74	0.01	0.01	0.01
4UAN21.2	10	6.18	0.01	0.01	0.01
4UAN22.2	11	6.50	0.01	0.01	0.01
6UAN23.4	12	6.84	0.01	0.01	0.01
6UAN24.6	13	7.22	0.01	0.01	0.01
5UAN25.6	14	7.50	0.01	0.01	0.01
5UAN27.2	15	8.00	0.01	0.01	0.01
6UAN28.6	16	8.36	0.01	0.01	0.01
7UAN31.6	17	9.24	0.01	0.01	0.01
8UAN34.4	18	10.12	0.01	0.01	0.01
8UAN39.2	19	11.50	0.01	0.01	0.01
8UAN57.4	20	16.88	0.01	0.01	0.01

1 6

168100 24 4 29000. 11600.

1	0.	0.	0.
2	240.	0.	0.
3	384.	0.	0.
4	720.	0.	0.
5	0.	144.	0.
6	240.	144.	0.
7	384.	144.	0.
8	720.	144.	0.
9	0.	288.	0.
10	240.	288.	0.
11	384.	288.	0.
12	720.	288.	0.
13	0.	432.	0.
14	240.	432.	0.
15	384.	432.	0.
16	720.	432.	0.

17	0.	576.	0.
18	240.	576.	0.
19	384.	576.	0.
20	720.	576.	0.
21	0.	720.	0.
22	240.	720.	0.
23	384.	720.	0.
24	720.	720.	0.
25	0.	864.	0.
26	240.	864.	0.
27	384.	864.	0.
28	720.	864.	0.
29	0.	1008.	0.
30	240.	1008.	0.
31	384.	1008.	0.
32	720.	1008.	0.
33	0.	1152.	0.
34	240.	1152.	0.
35	384.	1152.	0.
36	720.	1152.	0.
37	0.	1296.	0.
38	240.	1296.	0.
39	384.	1296.	0.
40	720.	1296.	0.
41	0.	1440.	0.
42	240.	1440.	0.
43	384.	1440.	0.
44	720.	1440.	0.
45	0.	1584.	0.
46	240.	1584.	0.
47	384.	1584.	0.
48	720.	1584.	0.
49	0.	1728.	0.
50	240.	1728.	0.
51	384.	1728.	0.

52	720.	1728.	0.
53	0.	1872.	0.
54	240.	1872.	0.
55	384.	1872.	0.
56	720.	1872.	0.
57	0.	2016.	0.
58	240.	2016.	0.
59	384.	2016.	0.
60	720.	2016.	0.
61	0.	2160.	0.
62	240.	2160.	0.
63	384.	2160.	0.
64	720.	2160.	0.
65	0.	2304.	0.
66	240.	2304.	0.
67	384.	2304.	0.
68	720.	2304.	0.
69	0.	2448.	0.
70	240.	2448.	0.
71	384.	2448.	0.
72	720.	2448.	0.
73	0.	2592.	0.
74	240.	2592.	0.
75	384.	2592.	0.
76	720.	2592.	0.
77	0.	2736.	0.
78	240.	2736.	0.
79	384.	2736.	0.
80	720.	2736.	0.
81	0.	2880.	0.
82	240.	2880.	0.
83	384.	2880.	0.
84	720.	2880.	0.
85	0.	3024.	0.
86	240.	3024.	0.

87	384.	3024.	0.
88	720.	3024.	0.
89	0.	3168.	0.
90	240.	3163.	0.
91	364.	3168.	0.
92	720.	3168.	0.
93	0.	3312.	0.
94	240.	3312.	0.
95	384.	3312.	0.
96	720.	3312.	0.
97	0.	3456.	0.
98	240.	3456.	0.
99	384.	3456.	0.
100	720.	3456.	0.

1	1	5	0	16	37	490.00	0.20
2	2	6	0	16	42	490.00	0.20
3	3	7	0	16	44	490.00	0.20
4	4	8	0	16	40	490.00	0.20
5	5	6	0	17	15	490.00	0.20
6	6	7	0	17	26	490.00	0.20
7	7	8	0	17	19	490.00	0.20
8	5	9	0	16	37	490.00	0.20
9	6	10	0	16	42	490.00	0.20
10	7	11	0	16	44	490.00	0.20
11	8	12	0	16	40	490.00	0.20
12	9	10	0	17	15	490.00	0.20
13	10	11	0	17	25	490.00	0.20
14	11	12	0	17	19	490.00	0.20
15	9	13	0	16	36	490.00	0.20
16	10	14	0	16	40	490.00	0.20
17	11	15	0	16	42	490.00	0.20
18	12	16	0	16	39	490.00	0.20
19	13	14	0	17	15	490.00	0.20
20	14	15	0	17	24	490.00	0.20
21	15	16	0	17	19	490.00	0.20

22	13	17	0	16	36	490.00	0.20
23	14	18	0	16	40	490.00	0.20
24	15	19	0	16	42	490.00	0.20
25	16	20	0	16	39	490.00	0.20
26	17	18	0	17	15	490.00	0.20
27	18	19	0	17	23	490.00	0.20
28	19	20	0	17	19	490.00	0.20
29	17	21	0	16	34	490.00	0.20
30	18	22	0	16	37	490.00	0.20
31	19	23	0	16	40	490.00	0.20
32	20	24	0	16	37	490.00	0.20
33	21	22	0	17	15	490.00	0.20
34	22	23	0	17	23	490.00	0.20
35	23	24	0	17	19	490.00	0.20
36	21	25	0	16	34	490.00	0.20
37	22	26	0	16	37	490.00	0.20
38	23	27	0	16	40	490.00	0.20
39	24	28	0	16	37	490.00	0.20
40	25	26	0	17	15	490.00	0.20
41	26	27	0	17	22	490.00	0.20
42	27	28	0	17	19	490.00	0.20
43	25	29	0	16	32	490.00	0.20
44	26	30	0	16	35	490.00	0.20
45	27	31	0	16	37	490.00	0.20
46	28	32	0	16	36	490.00	0.20
47	29	30	0	17	15	490.00	0.20
48	30	31	0	17	21	490.00	0.20
49	31	32	0	17	19	490.00	0.20
50	29	33	0	16	32	490.00	0.20
51	30	34	0	16	35	490.00	0.20
52	31	35	0	16	37	490.00	0.20
53	32	36	0	16	36	490.00	0.20
54	33	34	0	17	15	490.00	0.20
55	34	35	0	17	21	490.00	0.20
56	35	36	0	17	19	490.00	0.20

57	33	37	0	16	29	490.00	0.20
58	34	38	0	16	31	490.00	0.20
59	35	39	0	16	35	490.00	0.20
60	36	40	0	16	34	490.00	0.20
61	37	38	0	17	15	490.00	0.20
62	38	39	0	17	20	490.00	0.20
63	39	40	0	17	19	490.00	0.20
64	37	41	0	16	29	490.00	0.20
65	38	42	0	16	31	490.00	0.20
66	39	43	0	16	35	490.00	0.20
67	40	44	0	16	34	490.00	0.20
68	41	42	0	17	15	490.00	0.20
69	42	43	0	17	19	490.00	0.20
70	43	44	0	17	19	490.00	0.20
71	41	45	0	16	26	490.00	0.20
72	42	46	0	16	27	490.00	0.20
73	43	47	0	16	31	490.00	0.20
74	44	48	0	16	31	490.00	0.20
75	45	46	0	17	15	490.00	0.20
76	46	47	0	17	18	490.00	0.20
77	47	48	0	17	19	490.00	0.20
78	45	49	0	16	26	490.00	0.20
79	46	50	0	16	27	490.00	0.20
80	47	51	0	16	31	490.00	0.20
81	48	52	0	16	31	490.00	0.20
82	49	50	0	17	15	490.00	0.20
83	50	51	0	17	16	490.00	0.20
84	51	52	0	17	19	490.00	0.20
85	49	53	0	16	23	490.00	0.20
86	50	54	0	16	23	490.00	0.20
87	51	55	0	16	28	490.00	0.20
88	52	56	0	16	28	490.00	0.20
89	53	54	0	17	15	490.00	0.20
90	54	55	0	17	14	490.00	0.20
91	55	56	0	17	19	490.00	0.20

92	53	57	0	16	23	490.00	0.20
93	54	58	0	16	23	490.00	0.20
94	55	59	0	16	28	490.00	0.20
95	56	60	0	16	28	490.00	0.20
96	57	58	0	17	15	490.00	0.20
97	58	59	0	17	10	490.00	0.20
98	59	60	0	17	19	490.00	0.20
99	57	61	0	16	20	490.00	0.20
100	58	62	0	16	20	490.00	0.20
101	59	63	0	16	24	490.00	0.20
102	60	64	0	16	25	490.00	0.20
103	61	62	0	17	15	490.00	0.20
104	62	63	0	17	9	490.00	0.20
105	63	64	0	17	19	490.00	0.20
106	61	65	0	16	20	490.00	0.20
107	62	66	0	16	20	490.00	0.20
108	63	67	0	16	24	490.00	0.20
109	64	68	0	16	25	490.00	0.20
110	65	66	0	17	15	490.00	0.20
111	66	67	0	17	9	490.00	0.20
112	67	68	0	17	18	490.00	0.20
113	65	69	0	16	18	490.00	0.20
114	66	70	0	16	19	490.00	0.20
115	67	71	0	16	21	490.00	0.20
116	68	72	0	16	21	490.00	0.20
117	69	70	0	17	15	490.00	0.20
118	70	71	0	17	9	490.00	0.20
119	71	72	0	17	17	490.00	0.20
120	69	73	0	16	18	490.00	0.20
121	70	74	0	16	18	490.00	0.20
122	71	75	0	16	21	490.00	0.20
123	72	76	0	16	21	490.00	0.20
124	73	74	0	17	15	490.00	0.20
125	74	75	0	17	9	490.00	0.20
126	75	76	0	17	17	490.00	0.20

127	73	77	0	16	16	490.00	0.20
128	74	78	0	16	16	490.00	0.20
129	75	79	0	16	18	490.00	0.20
130	76	80	0	16	18	490.00	0.20
131	77	78	0	17	14	490.00	0.20
132	78	79	0	17	9	490.00	0.20
133	79	80	0	17	17	490.00	0.20
134	77	81	0	16	16	490.00	0.20
135	78	82	0	16	16	490.00	0.20
136	79	83	0	16	19	490.00	0.20
137	80	84	0	16	18	490.00	0.20
138	81	82	0	17	13	490.00	0.20
139	82	83	0	17	8	490.00	0.20
140	83	84	0	17	17	490.00	0.20
141	81	85	0	16	12	490.00	0.20
142	82	86	0	16	10	490.00	0.20
143	83	87	0	16	13	490.00	0.20
144	84	88	0	16	14	490.00	0.20
145	85	86	0	17	13	490.00	0.20
146	86	87	0	17	7	490.00	0.20
147	87	88	0	17	17	490.00	0.20
148	85	89	0	16	12	490.00	0.20
149	86	90	0	16	10	490.00	0.20
150	87	91	0	16	13	490.00	0.20
151	88	92	0	16	14	490.00	0.20
152	89	90	0	17	13	490.00	0.20
153	90	91	0	17	7	490.00	0.20
154	91	92	0	17	17	490.00	0.20
155	89	93	0	16	7	490.00	0.20
156	90	94	0	16	6	490.00	0.20
157	91	95	0	16	9	490.00	0.20
158	92	96	0	16	10	490.00	0.20
159	93	94	0	17	13	490.00	0.20
160	94	95	0	17	7	490.00	0.20
161	95	96	0	17	17	490.00	0.20

162	93	97	0	16	7	490.00	0.20
163	94	98	0	16	6	490.00	0.20
164	95	99	0	16	9	490.00	0.20
165	96	100	0	16	10	490.00	0.20
166	97	98	0	17	9	490.00	0.20
167	98	99	0	17	4	490.00	0.20
168	99	100	0	17	15	490.00	0.20

1	1	1	1	1	1	1
2	1	1	1	1	1	1
3	1	1	1	1	1	1
4	1	1	1	1	1	1

24 0

5	5.760	0.000	0.000	0.000	0.000	0.000
9	5.760	0.000	0.000	0.000	0.000	0.000
13	5.760	0.000	0.000	0.000	0.000	0.000
17	5.760	0.000	0.000	0.000	0.000	0.000
21	5.760	0.000	0.000	0.000	0.000	0.000
25	5.760	0.000	0.000	0.000	0.000	0.000
29	5.760	0.000	0.000	0.000	0.000	0.000
33	5.760	0.000	0.000	0.000	0.000	0.000
37	5.760	0.000	0.000	0.000	0.000	0.000
41	5.760	0.000	0.000	0.000	0.000	0.000
45	5.760	0.000	0.000	0.000	0.000	0.000
49	5.760	0.000	0.000	0.000	0.000	0.000
53	5.760	0.000	0.000	0.000	0.000	0.000
57	5.760	0.000	0.000	0.000	0.000	0.000
61	5.760	0.000	0.000	0.000	0.000	0.000
65	5.760	0.000	0.000	0.000	0.000	0.000
69	5.760	0.000	0.000	0.000	0.000	0.000
73	5.760	0.000	0.000	0.000	0.000	0.000
77	5.760	0.000	0.000	0.000	0.000	0.000
81	5.760	0.000	0.000	0.000	0.000	0.000
85	5.760	0.000	0.000	0.000	0.000	0.000
89	5.760	0.000	0.000	0.000	0.000	0.000
93	5.760	0.000	0.000	0.000	0.000	0.000

97	4.800	0.000	0.000	0.000	0.000	0.000
24	0					
8	-5.760	0.000	0.000	0.000	0.000	0.000
12	-5.760	0.000	0.000	0.000	0.000	0.000
16	-5.760	0.000	0.000	0.000	0.000	0.000
20	-5.760	0.000	0.000	0.000	0.000	0.000
24	-5.760	0.000	0.000	0.000	0.000	0.000
28	-5.760	0.000	0.000	0.000	0.000	0.000
32	-5.760	0.000	0.000	0.000	0.000	0.000
36	-5.760	0.000	0.000	0.000	0.000	0.000
40	-5.760	0.000	0.000	0.000	0.000	0.000
44	-5.760	0.000	0.000	0.000	0.000	0.000
48	-5.760	0.000	0.000	0.000	0.000	0.000
52	-5.760	0.000	0.000	0.000	0.000	0.000
56	-5.760	0.000	0.000	0.000	0.000	0.000
60	-5.760	0.000	0.000	0.000	0.000	0.000
64	-5.760	0.000	0.000	0.000	0.000	0.000
68	-5.760	0.000	0.000	0.000	0.000	0.000
72	-5.760	0.000	0.000	0.000	0.000	0.000
76	-5.760	0.000	0.000	0.000	0.000	0.000
80	-5.760	0.000	0.000	0.000	0.000	0.000
84	-5.760	0.000	0.000	0.000	0.000	0.000
88	-5.760	0.000	0.000	0.000	0.000	0.000
92	-5.760	0.000	0.000	0.000	0.000	0.000
96	-5.760	0.000	0.000	0.000	0.000	0.000
100	-4.800	0.000	0.000	0.000	0.000	0.000
1	0					
97	100.000	0.000	0.000	0.000	0.000	0.000
1	0					
100	-100.000	0.000	0.000	0.000	0.000	0.000
1	0					
69	100.000	0.000	0.000	0.000	0.000	0.000

```
*DECK 11 STORY L-SHAPED BRACED SPACE FRAME EXAMPLE BW2
SPACE FRAME --- 11 STORIES BRACED (TENSION ONLY)
   48    38    20     0    10     2   250     1     0     0     4     2  100.00  0.00 10.00
    3     1     1
   94     0
       1.00        0.00        0.00        0.00
       3.60     1
    4     1     2
   96     0
       0.00        0.00        1.00        0.00
       3.60     1
   6WF20        1         5.90        0.24       13.30       41.70
   8WF24        2         7.06        0.34       18.20       62.50
   8WF28        3         8.23        0.53       21.60       97.80
   8WF31        4         9.12        0.53       37.00      109.70
   8WF35        5        10.30        0.77       42.50      126.50
  10WF39        6        11.48        0.97       44.90      209.70
  12WF40        7        11.77        0.96       44.10      310.10
  14WF43        8        12.65        1.05       45.10      429.10
  14WF48        9        14.11        1.44       51.30      484.90
  14WF53       10        15.59        1.93       57.50      542.10
  12WF58       11        17.06        2.10      107.40      476.10
  14WF61       12        17.94        2.19      107.30      641.50
  14WF74       13        21.76        3.86      133.50      796.80
  14WF78       14        22.94        3.52      206.90      851.20
  12WF79       15        23.22        3.85      216.40      663.00
  14WF84       16        24.71        4.41      225.50      928.40
  12WF99       17        29.09        7.45      276.20      858.50
 14WF111       18        32.65        7.48      454.90     1266.50
 14WF119       19        34.99        9.20      491.60     1373.10
 14WF127       20        37.33       11.10      527.60     1476.70
 14WF136       21        39.93       13.50      567.70     1593.00
 14WF142       22        41.85       14.20      660.10     1672.20
 14WF150       23        44.08       16.70      702.50     1786.90
```

14WF158	24	46.47	19.50	745.00	1900.60
14WF167	25	49.09	22.80	790.20	2020.80
14WF176	26	51.73	25.50	837.90	2149.60
14WF184	27	54.07	30.30	882.70	2274.80
14WF193	28	56.73	34.70	930.10	2402.40
14WF202	29	59.39	39.60	979.70	2538.80
14WF211	30	62.07	44.80	1028.60	2671.40
14WF219	31	64.36	49.90	1073.20	2798.20
14WF228	32	67.06	55.20	1124.60	2942.40
14WF237	33	69.69	62.60	1174.80	3080.90
14WF246	34	72.33	69.70	1226.60	3228.90
14WF264	35	77.63	85.30	1331.20	3526.00
14WF287	36	84.37	108.00	1466.90	3912.10
14WF314	37	92.30	140.00	1631.40	4399.40
14WF320	38	94.12	137.00	1635.10	4141.70
14WF342	39	100.59	178.00	1806.90	4911.50
14WF370	40	108.73	222.00	1986.00	5454.20
14WF398	41	116.93	272.00	2169.70	6013.70
14WF426	42	125.25	330.00	2359.50	6610.30
14WF455	43	133.73	396.00	2561.20	7214.90
14WF500	44	146.95	514.00	2832.70	8234.10
14WF550	45	161.75	670.00	3256.70	9443.10
14WF605	46	177.85	869.00	3680.90	10842.30
14WF665	47	195.51	1120.00	4166.20	12477.70
14WF730	48	214.65	1450.00	4716.80	14371.40
6JR4.4	1	1.30	0.01	0.17	7.30
8JR6.5	2	1.92	0.02	0.34	18.70
10JR9	3	2.64	0.03	0.61	39.00
12JR11.8	4	3.45	0.04	0.98	72.00
10B15	5	4.40	0.10	2.79	68.80
12B16.5	6	4.86	0.11	2.79	105.30
14B17.2	7	5.05	0.11	2.65	147.30
14B22	8	6.47	0.21	6.40	197.40
16B26	9	7.65	0.26	8.71	298.10
14WF30	10	8.81	0.38	17.50	289.60

16B31	11	9.12	0.46	11.57	372.50
14WF34	12	10.00	0.57	21.30	339.20
16WF36	13	10.59	0.55	22.10	446.30
16WF40	14	11.77	0.79	26.50	515.50
18WF45	15	13.24	0.89	31.90	704.50
18WF50	16	14.71	1.25	37.20	800.60
21WF55	17	16.18	1.24	44.00	1140.70
21WF62	18	18.23	1.83	53.10	1326.80
24WF68	19	20.00	1.86	63.60	1814.50
24WF76	20	22.37	2.70	76.50	2096.40
27WF84	21	24.71	2.79	95.70	2824.80
27WF94	22	27.65	4.06	115.10	3266.70
30WF99	23	29.11	3.78	116.90	3988.60
30WF108	24	31.77	5.02	135.10	4461.00
30WF116	25	34.13	6.43	153.20	4919.10
33WF118	26	34.71	5.32	170.30	5886.90
33WF130	27	38.26	7.37	201.40	6699.00
36WF135	28	39.70	7.03	207.10	7796.10
36WF150	29	44.16	10.10	250.40	9012.10
36WF160	30	47.09	12.40	275.40	9738.80
36WF170	31	49.98	15.10	300.60	10470.00
36WF182	32	53.54	18.40	327.70	11281.50
36WF194	33	57.11	22.30	355.40	12103.40
36WF230	34	67.73	28.60	870.90	14988.40
36WF245	35	72.03	34.70	944.70	16092.20
36WF260	36	76.56	41.60	1020.60	17233.80
36WF280	37	82.32	52.60	1127.50	18819.30
36WF300	38	88.17	64.20	1225.20	20290.20
3UAN6.1	1	1.80	0.01	0.01	0.01
3UAN9.0	2	2.62	0.01	0.01	0.01
4UAN11.6	3	3.38	0.01	0.01	0.01
3UAN13.2	4	3.84	0.01	0.01	0.01
4UAN14.4	5	4.18	0.01	0.01	0.01
4UAN15.4	6	4.50	0.01	0.01	0.01
4UAN17.0	7	4.96	0.01	0.01	0.01

4UAN18.2	8	5.34	0.01	0.01	0.01
4UAN19.6	9	5.74	0.01	0.01	0.01
4UAN21.2	10	6.18	0.01	0.01	0.01
4UAN22.2	11	6.50	0.01	0.01	0.01
6UAN23.4	12	6.84	0.01	0.01	0.01
6UAN24.6	13	7.22	0.01	0.01	0.01
5UAN25.6	14	7.50	0.01	0.01	0.01
5UAN27.2	15	8.00	0.01	0.01	0.01
6UAN28.6	16	8.36	0.01	0.01	0.01
7UAN31.6	17	9.24	0.01	0.01	0.01
8UAN34.4	18	10.12	0.01	0.01	0.01
8UAN39.2	19	11.50	0.01	0.01	0.01
8UAN57.4	20	16.88	0.01	0.01	0.01

1 6

264 96 48 8 29000. 11600.

1	0.	0.	0.
2	240.	0.	0.
3	600.	0.	0.
4	0.	0.	240.
5	240.	0.	240.
6	600.	0.	240.
7	0.	0.	600.
8	240.	0.	600.
9	0.	180.	0.
10	240.	180.	0.
11	600.	180.	0.
12	0.	180.	240.
13	240.	180.	240.
14	600.	180.	240.
15	0.	180.	600.
16	240.	180.	600.
17	0.	360.	0.
18	240.	360.	0.
19	600.	360.	0.
20	0.	360.	240.

21	240.	360.	240.
22	600.	360.	240.
23	0.	360.	600.
24	240.	360.	600.
25	0.	480.	0.
26	240.	480.	0.
27	600.	480.	0.
28	0.	480.	240.
29	240.	480.	240.
30	600.	480.	240.
31	0.	480.	600.
32	240.	480.	600.
33	0.	600.	0.
34	240.	600.	0.
35	600.	600.	0.
36	0.	600.	240.
37	240.	600.	240.
38	600.	600.	240.
39	0.	600.	600.
40	240.	600.	600.
41	0.	720.	0.
42	240.	720.	0.
43	600.	720.	0.
44	0.	720.	240.
45	240.	720.	240.
46	600.	720.	240.
47	0.	720.	600.
48	240.	720.	600.
49	0.	840.	0.
50	240.	840.	0.
51	600.	840.	0.
52	0.	840.	240.
53	240.	840.	240.
54	600.	840.	240.
55	0.	840.	600.

56	240.	840.	600.
57	0.	960.	0.
58	240.	960.	0.
59	600.	960.	0.
60	0.	960.	240.
61	240.	960.	240.
62	600.	960.	240.
63	0.	960.	600.
64	240.	960.	600.
65	0.	1080.	0.
66	240.	1080.	0.
67	600.	1080.	0.
68	0.	1080.	240.
69	240.	1080.	240.
70	600.	1080.	240.
71	0.	1080.	600.
72	240.	1080.	600.
73	0.	1200.	0.
74	240.	1200.	0.
75	600.	1200.	0.
76	0.	1200.	240.
77	240.	1200.	240.
78	600.	1200.	240.
79	0.	1200.	600.
80	240.	1200.	600.
81	0.	1320.	0.
82	240.	1320.	0.
83	600.	1320.	0.
84	0.	1320.	240.
85	240.	1320.	240.
86	600.	1320.	240.
87	0.	1320.	600.
88	240.	1320.	600.
89	0.	1440.	0.
90	240.	1440.	0.

```
91    600.  1440.      0.
92      0.  1440.    240.
93    240.  1440.    240.
94    600.  1440.    240.
95      0.  1440.    600.
96    240.  1440.    600.
    1     1     9    0    16    23    490.00     0.20
    2     2    10    1    16    23    490.00     0.20
 2    240.      0.   -10.
    3     3    11    1    16    23    490.00     0.20
 3    600.      0.   -10.
    4     4    12    1    16    23    490.00     0.20
 4      0.      0.   -10.
    5     5    13    0    16    23    490.00     0.20
    6     6    14    1    16    23    490.00     0.20
 6    600.      0.   -10.
    7     7    15    0    16    23    490.00     0.20
    8     8    16    0    16    23    490.00     0.20
    9     9    10    0    17    15    490.00     0.20
   10    10    11    0    17    15    490.00     0.20
   11     9    12    0    17    15    490.00     0.20
   12    10    13    0    17    15    490.00     0.20
   13    11    14    0    17    15    490.00     0.20
   14    12    13    0    17    15    490.00     0.20
   15    13    14    0    17    15    490.00     0.20
   16    12    15    0    17    15    490.00     0.20
   17    13    16    0    17    15    490.00     0.20
   18    15    16    0    17    15    490.00     0.20
   19     9    17    0    16    23    490.00     0.20
   20    10    18    1    16    23    490.00     0.20
20    240.    180.   -10.
   21    11    19    1    16    23    490.00     0.20
21    600.    180.   -10.
   22    12    20    1    16    23    490.00     0.20
22      0.    180.   -10.
```

23	13	21	0	16	23	490.00	0.20
24	14	22	1	16	23	490.00	0.20
24	600.	180.	-10.				
25	15	23	0	16	23	490.00	0.20
26	16	24	0	16	23	490.00	0.20
27	17	16	0	17	15	490.00	0.20
28	18	19	0	17	15	490.00	0.20
29	17	20	0	17	15	490.00	0.20
30	18	21	0	17	15	490.00	0.20
31	19	22	0	17	15	490.00	0.20
32	20	21	0	17	15	490.00	0.20
33	21	22	0	17	15	490.00	0.20
34	20	23	0	17	15	490.00	0.20
35	21	24	0	17	15	490.00	0.20
36	23	24	0	17	15	490.00	0.20
37	17	25	0	16	18	490.00	0.20
38	18	26	1	16	18	490.00	0.20
38	240.	360.	-10.				
39	19	27	1	16	18	490.00	0.20
39	600.	360.	-10.				
40	20	28	1	16	18	490.00	0.20
40	0.	360.	-10.				
41	21	29	0	16	18	490.00	0.20
42	22	30	1	16	18	490.00	0.20
42	600.	360.	-10.				
43	23	31	0	16	18	490.00	0.20
44	24	32	0	16	18	490.00	0.20
45	25	26	0	17	15	490.00	0.20
46	26	27	0	17	15	490.00	0.20
47	25	28	0	17	15	490.00	0.20
48	26	29	0	17	15	490.00	0.20
49	27	30	0	17	15	490.00	0.20
50	28	29	0	17	15	490.00	0.20
51	29	30	0	17	15	490.00	0.20
52	28	31	0	17	15	490.00	0.20

53	29	32	0	17	15	490.00	0.20
54	31	32	0	17	15	490.00	0.20
55	25	33	0	16	18	490.00	0.20
56	26	34	1	16	18	490.00	0.20
56	240.	480.	-10.				
57	27	35	1	16	18	490.00	0.20
57	600.	480.	-10.				
58	28	36	1	16	18	490.00	0.20
58	0.	480.	-10.				
59	29	37	0	16	18	490.00	0.20
60	30	38	1	16	18	490.00	0.20
60	600.	480.	-10.				
61	31	39	0	16	18	490.00	0.20
62	32	40	0	16	18	490.00	0.20
63	33	34	0	17	15	490.00	0.20
64	34	35	0	17	15	490.00	0.20
65	33	36	0	17	15	490.00	0.20
66	34	37	0	17	15	490.00	0.20
67	35	38	0	17	15	490.00	0.20
68	36	37	0	17	15	490.00	0.20
69	37	38	0	17	15	490.00	0.20
70	36	39	0	17	15	490.00	0.20
71	37	40	0	17	15	490.00	0.20
72	39	40	0	17	15	490.00	0.20
73	33	41	0	16	13	490.00	0.20
74	34	42	1	16	13	490.00	0.20
74	240.	600.	-10.				
75	35	43	1	16	13	490.00	0.20
75	600.	600.	-10.				
76	36	44	1	16	13	490.00	0.20
76	0.	600.	-10.				
77	37	45	0	16	13	490.00	0.20
78	38	46	1	16	13	490.00	0.20
78	600.	600.	-10.				
79	39	47	0	16	13	490.00	0.20

80	40	48	0	16	13	490.00	0.20
81	41	42	0	17	13	490.00	0.20
82	42	43	0	17	13	490.00	0.20
83	41	44	0	17	13	490.00	0.20
84	42	45	0	17	13	490.00	0.20
85	43	46	0	17	13	490.00	0.20
86	44	45	0	17	13	490.00	0.20
87	45	46	0	17	13	490.00	0.20
88	44	47	0	17	13	490.00	0.20
89	45	48	0	17	13	490.00	0.20
90	47	48	0	17	13	490.00	0.20
91	41	49	0	16	13	490.00	0.20
92	42	50	1	16	13	490.00	0.20
92	240.	720.	-10.				
93	43	51	1	16	13	490.00	0.20
93	600.	720.	-10.				
94	44	52	1	16	13	490.00	0.20
94	0.	720.	-10.				
95	45	53	0	16	13	490.00	0.20
96	46	54	1	16	13	490.00	0.20
96	600.	720.	-10.				
97	47	55	0	16	13	490.00	0.20
98	48	56	0	16	13	490.00	0.20
99	49	50	0	17	13	490.00	0.20
100	50	51	0	17	13	490.00	0.20
101	49	52	0	17	13	490.00	0.20
102	50	53	0	17	13	490.00	0.20
103	51	54	0	17	13	490.00	0.20
104	52	53	0	17	13	490.00	0.20
105	53	54	0	17	13	490.00	0.20
106	52	55	0	17	13	490.00	0.20
107	53	56	0	17	13	490.00	0.20
108	55	56	0	17	13	490.00	0.20
109	49	57	0	16	7	490.00	0.20
110	50	58	1	16	7	490.00	0.20

110	240.	840.	-10.				
111	51	59	1	16	7	490.00	0.20
111	600.	840.	-10.				
112	52	60	1	16	7	490.00	0.20
112	0.	840.	-10.				
113	53	61	0	16	7	490.00	0.20
114	54	62	1	16	7	490.00	0.20
114	600.	840.	-10.				
115	55	63	0	16	7	490.00	0.20
116	56	64	0	16	7	490.00	0.20
117	57	58	0	17	13	490.00	0.20
118	58	59	0	17	13	490.00	0.20
119	57	60	0	17	13	490.00	0.20
120	58	61	0	17	13	490.00	0.20
121	59	62	0	17	13	490.00	0.20
122	60	61	0	17	13	490.00	0.20
123	61	62	0	17	13	490.00	0.20
124	60	63	0	17	13	490.00	0.20
125	61	64	0	17	13	490.00	0.20
126	63	64	0	17	13	490.00	0.20
127	57	65	0	16	7	490.00	0.20
128	58	66	1	16	7	490.00	0.20
128	240.	960.	-10.				
129	59	67	1	16	7	490.00	0.20
129	600.	960.	-10.				
130	60	68	1	16	7	490.00	0.20
130	0.	960.	-10.				
131	61	69	0	16	7	490.00	0.20
132	62	70	1	16	7	490.00	0.20
132	600.	960.	-10.				
133	63	71	0	16	7	490.00	0.20
134	64	72	0	16	7	490.00	0.20
135	65	66	0	17	13	490.00	0.20
136	66	67	0	17	13	490.00	0.20
137	65	68	0	17	13	490.00	0.20

138	66	69	0	17	13	490.00	0.20
139	67	70	0	17	13	490.00	0.20
140	68	69	0	17	13	490.00	0.20
141	69	70	0	17	13	490.00	0.20
142	68	71	0	17	13	490.00	0.20
143	69	72	0	17	13	490.00	0.20
144	71	72	0	17	13	490.00	0.20
145	65	73	0	16	4	490.00	0.20
146	66	74	1	16	4	490.00	0.20
146	240.	1080.	-10.				
147	67	75	1	16	4	490.00	0.20
147	600.	1080.	-10.				
148	68	76	1	16	4	490.00	0.20
148	0.	1080.	-10.				
149	69	77	0	16	4	490.00	0.20
150	70	78	1	16	4	490.00	0.20
150	600.	1080.	-10.				
151	71	79	0	16	4	490.00	0.20
152	72	80	0	16	4	490.00	0.20
153	73	74	0	17	9	490.00	0.20
154	74	75	0	17	9	490.00	0.20
155	73	76	0	17	9	490.00	0.20
156	74	77	0	17	9	490.00	0.20
157	75	78	0	17	9	490.00	0.20
158	76	77	0	17	9	490.00	0.20
159	77	78	0	17	9	490.00	0.20
160	76	79	0	17	9	490.00	0.20
161	77	80	0	17	9	490.00	0.20
162	79	80	0	17	9	490.00	0.20
163	73	81	0	16	4	490.00	0.20
164	74	82	1	16	4	490.00	0.20
164	240.	1200.	-10.				
165	75	83	1	16	4	490.00	0.20
165	600.	1200.	-10.				
166	76	84	1	16	4	490.00	0.20

166	0.	1200.	-10.					
	167	77	85	0	16	4	490.00	0.20
	168	78	86	1	16	4	490.00	0.20
168	600.	1200.	-10.					
	169	79	87	0	16	4	490.00	0.20
	170	80	88	0	16	4	490.00	0.20
	171	81	82	0	17	9	490.00	0.20
	172	82	83	0	17	9	490.00	0.20
	173	81	84	0	17	9	490.00	0.20
	174	82	85	0	17	9	490.00	0.20
	175	83	86	0	17	9	490.00	0.20
	176	84	85	0	17	9	490.00	0.20
	177	85	86	0	17	9	490.00	0.20
	178	84	87	0	17	9	490.00	0.20
	179	85	88	0	17	9	490.00	0.20
	180	87	88	0	17	9	490.00	0.20
	181	81	89	0	16	2	490.00	0.20
	182	82	90	1	16	2	490.00	0.20
182	240.	1320.	-10.					
	183	83	91	1	16	2	490.00	0.20
183	600.	1320.	-10.					
	184	84	92	1	16	2	490.00	0.20
184	0.	1320.	-10.					
	185	85	93	0	16	2	490.00	0.20
	186	86	94	1	16	2	490.00	0.20
186	600.	1320.	-10.					
	187	87	95	0	16	2	490.00	0.20
	188	88	96	0	16	2	490.00	0.20
	189	89	90	0	17	9	490.00	0.20
	190	90	91	0	17	9	490.00	0.20
	191	89	92	0	17	9	490.00	0.20
	192	90	93	0	17	9	490.00	0.20
	193	91	94	0	17	9	490.00	0.20
	194	92	93	0	17	9	490.00	0.20
	195	93	94	0	17	9	490.00	0.20

196	92	95	0	17	9	490.00	0.20
197	93	96	0	17	9	490.00	0.20
198	95	96	0	17	9	490.00	0.20
199	4	15	0	18	3	490.00	0.20
200	12	23	0	18	3	490.00	0.20
201	20	31	0	18	3	490.00	0.20
202	28	39	0	18	3	490.00	0.20
203	36	47	0	18	2	490.00	0.20
204	44	55	0	18	2	490.00	0.20
205	52	63	0	18	2	490.00	0.20
206	60	71	0	18	2	490.00	0.20
207	68	79	0	18	1	490.00	0.20
208	76	87	0	18	1	490.00	0.20
209	84	95	0	18	1	490.00	0.20
210	5	16	0	18	3	490.00	0.20
211	13	24	0	18	3	490.00	0.20
212	21	32	0	18	3	490.00	0.20
213	29	40	0	18	3	490.00	0.20
214	37	48	0	18	2	490.00	0.20
215	45	56	0	18	2	490.00	0.20
216	53	64	0	18	2	490.00	0.20
217	61	72	0	18	2	490.00	0.20
218	69	80	0	18	1	490.00	0.20
219	77	88	0	18	1	490.00	0.20
220	85	96	0	18	1	490.00	0.20
221	3	14	0	18	3	490.00	0.20
222	11	22	0	18	3	490.00	0.20
223	19	30	0	18	3	490.00	0.20
224	27	38	0	18	3	490.00	0.20
225	35	46	0	18	2	490.00	0.20
226	43	54	0	18	2	490.00	0.20
227	51	62	0	18	2	490.00	0.20
228	59	70	0	18	2	490.00	0.20
229	67	78	0	18	1	490.00	0.20
230	75	86	0	18	1	490.00	0.20

231	83	94	0	18	1	490.00	0.20
232	2	11	0	18	3	490.00	0.20
233	10	19	0	18	3	490.00	0.20
234	18	27	0	18	3	490.00	0.20
235	26	35	0	18	3	490.00	0.20
236	34	43	0	18	2	490.00	0.20
237	42	51	0	18	2	490.00	0.20
238	50	59	0	18	2	490.00	0.20
239	58	67	0	18	2	490.00	0.20
240	66	75	0	18	1	490.00	0.20
241	74	83	0	18	1	490.00	0.20
242	82	91	0	18	1	490.00	0.20
243	5	14	0	18	3	490.00	0.20
244	13	22	0	18	3	490.00	0.20
245	21	30	0	18	3	490.00	0.20
246	29	38	0	18	3	490.00	0.20
247	37	46	0	18	2	490.00	0.20
248	45	54	0	18	2	490.00	0.20
249	53	62	0	18	2	490.00	0.20
250	61	70	0	18	2	490.00	0.20
251	69	78	0	18	1	490.00	0.20
252	77	86	0	18	1	490.00	0.20
253	85	94	0	18	1	490.00	0.20
254	7	16	0	18	3	490.00	0.20
255	15	24	0	18	3	490.00	0.20
256	23	32	0	18	3	490.00	0.20
257	31	40	0	18	3	490.00	0.20
258	39	48	0	18	2	490.00	0.20
259	47	56	0	18	2	490.00	0.20
260	55	64	0	18	2	490.00	0.20
261	63	72	0	18	2	490.00	0.20
262	71	80	0	18	1	490.00	0.20
263	79	88	0	18	1	490.00	0.20
264	87	96	0	18	1	490.00	0.20

1 1 1 1 1 1 1

2	1	1	1	1	1	1
3	1	1	1	1	1	1
4	1	1	1	1	1	1
5	1	1	1	1	1	1
6	1	1	1	1	1	1
7	1	1	1	1	1	1
8	1	1	1	1	1	1
33	0					

9	7.500	0.000	0.000	0.000	0.000	0.000
12	18.750	0.000	0.000	0.000	0.000	0.000
15	11.250	0.000	0.000	0.000	0.000	0.000
17	6.750	0.000	0.000	0.000	0.000	0.000
20	16.875	0.000	0.000	0.000	0.000	0.000
23	10.125	0.000	0.000	0.000	0.000	0.000
25	6.000	0.000	0.000	0.000	0.000	0.000
28	15.000	0.000	0.000	0.000	0.000	0.000
31	9.000	0.000	0.000	0.000	0.000	0.000
33	6.000	0.000	0.000	0.000	0.000	0.000
36	15.000	0.000	0.000	0.000	0.000	0.000
39	9.000	0.000	0.000	0.000	0.000	0.000
41	6.000	0.000	0.000	0.000	0.000	0.000
44	15.000	0.000	0.000	0.000	0.000	0.000
47	9.000	0.000	0.000	0.000	0.000	0.000
49	6.000	0.000	0.000	0.000	0.000	0.000
52	15.000	0.000	0.000	0.000	0.000	0.000
55	9.000	0.000	0.000	0.000	0.000	0.000
57	6.000	0.000	0.000	0.000	0.000	0.000
60	15.000	0.000	0.000	0.000	0.000	0.000
63	9.000	0.000	0.000	0.000	0.000	0.000
65	6.000	0.000	0.000	0.000	0.000	0.000
68	15.000	0.000	0.000	0.000	0.000	0.000
71	9.000	0.000	0.000	0.000	0.000	0.000
73	6.000	0.000	0.000	0.000	0.000	0.000
76	15.000	0.000	0.000	0.000	0.000	0.000
79	9.000	0.000	0.000	0.000	0.000	0.000

81	6.000	0.000	0.000	0.000	0.000	0.000
84	15.000	0.000	0.000	0.000	0.000	0.000
87	9.000	0.000	0.000	0.000	0.000	0.000
89	4.500	0.000	0.000	0.000	0.000	0.000
92	11.250	0.000	0.000	0.000	0.000	0.000
95	6.750	0.000	0.000	0.000	0.000	0.000
33 0						
9	0.000	0.000	7.500	0.000	0.000	0.000
10	0.000	0.000	18.750	0.000	0.000	0.000
11	0.000	0.000	11.250	0.000	0.000	0.000
17	0.000	0.000	6.750	0.000	0.000	0.000
18	0.000	0.000	16.875	0.000	0.000	0.000
19	0.000	0.000	10.125	0.000	0.000	0.000
25	0.000	0.000	6.000	0.000	0.000	0.000
26	0.000	0.000	15.000	0.000	0.000	0.000
27	0.000	0.000	9.000	0.000	0.000	0.000
33	0.000	0.000	6.000	0.000	0.000	0.000
34	0.000	0.000	15.000	0.000	0.000	0.000
35	0.000	0.000	9.000	0.000	0.000	0.000
41	0.000	0.000	6.000	0.000	0.000	0.000
42	0.000	0.000	15.000	0.000	0.000	0.000
43	0.000	0.000	9.000	0.000	0.000	0.000
49	0.000	0.000	6.000	0.000	0.000	0.000
50	0.000	0.000	15.000	0.000	0.000	0.000
51	0.000	0.000	9.000	0.000	0.000	0.000
57	0.000	0.000	6.000	0.000	0.000	0.000
58	0.000	0.000	15.000	0.000	0.000	0.000
59	0.000	0.000	9.000	0.000	0.000	0.000
65	0.000	0.000	6.000	0.000	0.000	0.000
66	0.000	0.000	15.000	0.000	0.000	0.000
67	0.000	0.000	9.000	0.000	0.000	0.000
73	0.000	0.000	6.000	0.000	0.000	0.000
74	0.000	0.000	15.000	0.000	0.000	0.000
75	0.000	0.000	9.000	0.000	0.000	0.000
81	0.000	0.000	6.000	0.000	0.000	0.000

82	0.000	0.000	15.000	0.000	0.000	0.000
83	0.000	0.000	9.000	0.000	0.000	0.000
89	0.000	0.000	4.500	0.000	0.000	0.000
90	0.000	0.000	11.250	0.000	0.000	0.000
91	0.000	0.000	6.750	0.000	0.000	0.000
1	0					
92	100.000	0.000	0.000	0.000	0.000	0.000
1	0					
90	0.000	0.000	100.000	0.000	0.000	0.000

```
*DECK 2 STORY 1 BAY BY 1 BAY SPACE FRAME - KINEMATIC COND, IN-PLANE RIGIDITY.
 SPACE FRAME --- TWO STORY ONE BAY BY ONE BAY
   48    38    20     1     1     2   100     1     1     1     4     2   10.00   0.00  10.00
    3     1     1
    9     0
       1.00        0.00        0.00        0.00
       0.60     1
    4     1     2
    9     0
       0.00        0.00        1.00        0.00
       0.60     1
   6WF20        1        5.90        0.24       13.30       41.70
   8WF24        2        7.06        0.34       18.20       82.50
   8WF28        3        8.23        0.53       21.60       97.80
   8WF31        4        9.12        0.53       37.00      109.70
   8WF35        5       10.30        0.77       42.50      126.50
  10WF39        6       11.48        0.97       44.90      209.70
  12WF40        7       11.77        0.96       44.10      310.10
  14WF43        8       12.65        1.05       45.10      429.10
  14WF48        9       14.11        1.44       51.30      484.90
  14WF53       10       15.59        1.93       57.50      542.10
  12WF58       11       17.06        2.10      107.40      476.10
  14WF61       12       17.94        2.19      107.30      641.50
  14WF74       13       21.76        3.86      133.50      796.80
  14WF78       14       22.94        3.52      206.90      851.20
  12WF79       15       23.22        3.85      216.40      663.00
  14WF84       16       24.71        4.41      225.50      928.40
  12WF99       17       29.09        7.45      278.20      858.50
 14WF111       18       32.65        7.48      454.90     1266.50
 14WF119       19       34.99        9.20      491.80     1373.10
 14WF127       20       37.33       11.10      527.60     1476.70
 14WF136       21       39.98       13.50      567.70     1593.00
 14WF142       22       41.85       14.20      660.10     1672.20
 14WF150       23       44.08       16.70      702.50     1786.90
```

14WF158	24	46.47	19.50	745.00	1900.60
14WF167	25	49.09	22.80	790.20	2020.80
14WF176	26	51.73	26.50	837.90	2149.60
14WF184	27	54.07	30.30	882.70	2274.80
14WF193	28	56.73	34.70	930.10	2402.40
14WF202	29	59.39	39.60	979.70	2538.80
14WF211	30	62.07	44.80	1028.60	2671.40
14WF219	31	64.36	49.90	1073.20	2798.20
14WF228	32	67.06	56.20	1124.80	2942.40
14WF237	33	69.69	62.60	1174.80	3030.90
14WF246	34	72.33	69.70	1226.60	3228.90
14WF264	35	77.63	85.30	1331.20	3526.00
14WF287	36	84.37	105.00	1466.50	3912.10
14WF314	37	92.30	140.00	1631.40	4399.40
14WF320	38	94.12	137.00	1635.10	4141.70
14WF342	39	100.59	178.00	1806.90	4911.50
14WF370	40	108.78	222.00	1986.00	5454.20
14WF398	41	116.98	272.00	2169.70	6013.70
14WF426	42	125.25	330.00	2359.50	6610.30
14WF455	43	133.73	396.00	2561.20	7214.90
14WF500	44	146.95	514.00	2882.70	8234.10
14WF550	45	161.75	670.00	3256.70	9443.10
14WF605	46	177.85	869.00	3680.90	10842.30
14WF665	47	195.51	1120.00	4166.20	12477.70
14WF730	48	214.65	1450.00	4716.60	14371.40
6JR4.4	1	1.30	0.01	0.17	7.30
8JR6.5	2	1.92	0.02	0.34	18.70
10JR9	3	2.64	0.03	0.61	39.00
12JR11.8	4	3.45	0.04	0.98	72.00
10B15	5	4.40	0.10	2.79	68.80
12B16.5	6	4.86	0.11	2.79	105.30
14B17.2	7	5.05	0.11	2.65	147.30
14B22	8	6.47	0.21	6.40	197.40
16B26	9	7.65	0.26	8.71	298.10
14WF30	10	8.81	0.38	17.50	289.60

16B31	11	9.12	0.46	11.57	372.50
14WF34	12	10.00	0.57	21.30	339.20
16WF36	13	10.59	0.55	22.10	446.30
16WF40	14	11.77	0.79	26.50	515.50
18WF45	15	13.24	0.89	31.90	704.50
18WF50	16	14.71	1.25	37.20	800.60
21WF55	17	16.18	1.24	44.00	1140.70
21WF62	18	18.23	1.83	53.10	1326.80
24WF68	19	20.00	1.86	63.80	1814.50
24WF76	20	22.37	2.70	76.50	2096.40
27WF84	21	24.71	2.79	95.70	2824.80
27WF94	22	27.65	4.06	115.10	3266.70
30WF99	23	29.11	3.78	116.90	3988.60
30WF108	24	31.77	5.02	135.10	4461.00
30WF116	25	34.13	6.43	153.20	4919.10
33WF118	26	34.71	5.32	170.30	5886.90
33WF130	27	38.26	7.37	201.40	6699.00
36WF135	28	39.70	7.03	207.10	7796.10
36WF150	29	44.16	10.10	250.40	9012.10
36WF160	30	47.09	12.40	275.40	9738.80
36WF170	31	49.98	15.10	300.60	10470.00
36WF182	32	53.54	18.40	327.70	11281.50
36WF194	33	57.11	22.30	355.40	12103.40
36WF230	34	67.73	28.60	870.90	14988.40
36WF245	35	72.03	34.70	944.70	16092.20
36WF260	36	76.56	41.60	1020.60	17233.80
36WF280	37	82.32	52.60	1127.50	18819.30
36WF300	38	88.17	64.20	1225.20	20290.20
3UAN6.1	1	1.80	0.01	0.01	0.01
3UAN9.0	2	2.62	0.01	0.01	0.01
4UAN11.6	3	3.38	0.01	0.01	0.01
3UAN13.2	4	3.84	0.01	0.01	0.01
4UAN14.4	5	4.13	0.01	0.01	0.01
4UAN15.4	6	4.50	0.01	0.01	0.01
4UAN17.0	7	4.96	0.01	0.01	0.01

```
4UAN18.2      8       5.34      0.01      0.01      0.01
4UAN19.6      9       5.74      0.01      0.01      0.01
4UAN21.2     10       6.18      0.01      0.01      0.01
4UAN22.2     11       6.50      0.01      0.01      0.01
6UAN23.4     12       6.84      0.01      0.01      0.01
6UAN24.6     13       7.22      0.01      0.01      0.01
5UAN25.6     14       7.50      0.01      0.01      0.01
5UAN27.2     15       8.00      0.01      0.01      0.01
6UAN28.6     16       8.36      0.01      0.01      0.01
7UAN31.6     17       9.24      0.01      0.01      0.01
8UAN34.4     18      10.12      0.01      0.01      0.01
8UAN39.2     19      11.50      0.01      0.01      0.01
8UAN57.4     20      16.88      0.01      0.01      0.01
  1  6
 20 12 24  4 29000.11600.
  1     0.      0.      0.
  2   240.      0.      0.
  3   240.      0.    240.
  4     0.      0.    240.
  5     0.    120.      0.
  6   240.    120.      0.
  7   240.    120.    240.
  8     0.    120.    240.
  9     0.    240.      0.
 10   240.    240.      0.
 11   240.    240.    240.
 12     0.    240.    240.
      1    1    5    0   16    2    490.00      0.20
      2    2    6    0   16    2    490.00      0.20
      3    3    7    0   16    2    490.00      0.20
      4    4    8    0   16    2    490.00      0.20
      5    5    6    0   17    8    490.00      0.20
      6    6    7    0   17    8    490.00      0.20
      7    7    8    0   17    8    490.00      0.20
      8    5    8    0   17    8    490.00      0.20
```

```
   9    5    9    0   16    2    490.00      0.20
  10    6   10    0   16    2    490.00      0.20
  11    7   11    0   16    2    490.00      0.20
  12    8   12    0   16    2    490.00      0.20
  13    9   10    0   17    8    490.00      0.20
  14   10   11    0   17    8    490.00      0.20
  15   11   12    0   17    8    490.00      0.20
  16    9   12    0   17    8    490.00      0.20
  17    5    7    0   18    2    490.00      0.20
  18    6    8    0   18    2    490.00      0.20
  19    9   11    0   18    2    490.00      0.20
  20   10   12    0   18    2    490.00      0.20
 1  1  1  1  1  1  1
 2  1  1  1  1  1  1
 3  1  1  1  1  1  1
 4  1  1  1  1  1  1
 4  0
 5      5.000      0.000      0.000      0.000      0.000      0.000
 8      5.000      0.000      0.000      0.000      0.000      0.000
 9      5.000      0.000      0.000      0.000      0.000      0.000
12      5.000      0.000      0.000      0.000      0.000      0.000
 4  0
 5      0.000      0.000      5.000      0.000      0.000      0.000
 6      0.000      0.000      5.000      0.000      0.000      0.000
 9      0.000      0.000      5.000      0.000      0.000      0.000
10      0.000      0.000      5.000      0.000      0.000      0.000
 1  0
 9     10.000      0.000      0.000      0.000      0.000      0.000
 1  0
 9      0.000      0.000     10.000      0.000      0.000      0.000
   2     120.00     120.00
   1    4    1
   5   -8
   2    4    1
   9  -12
```